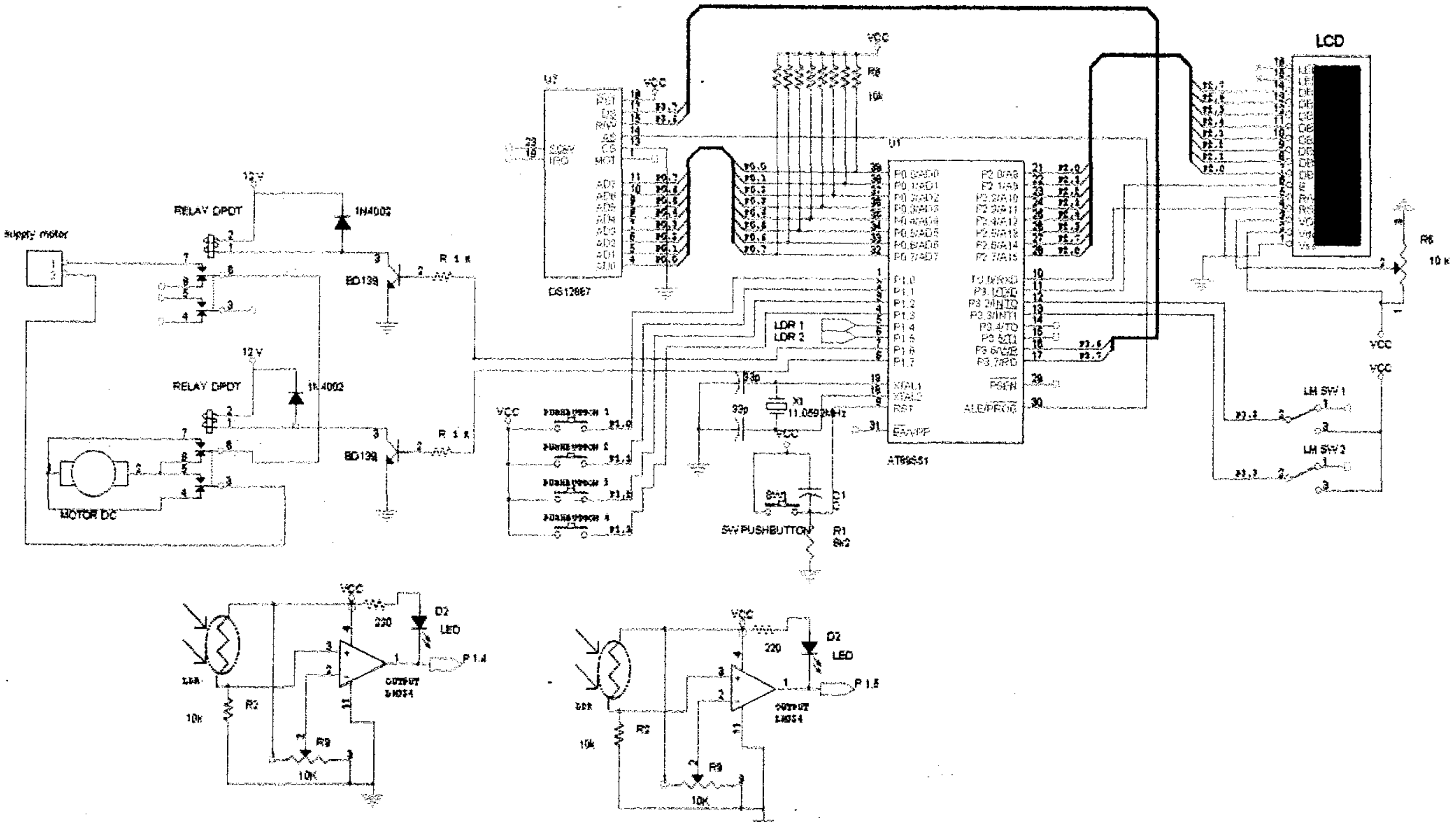
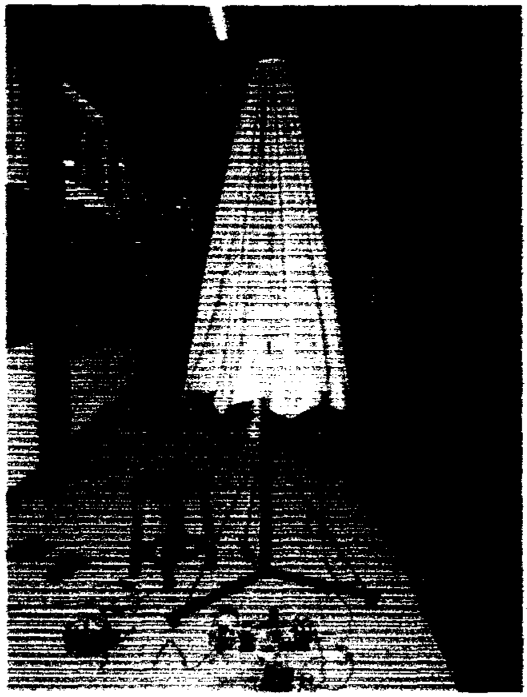
LAMPIRAN

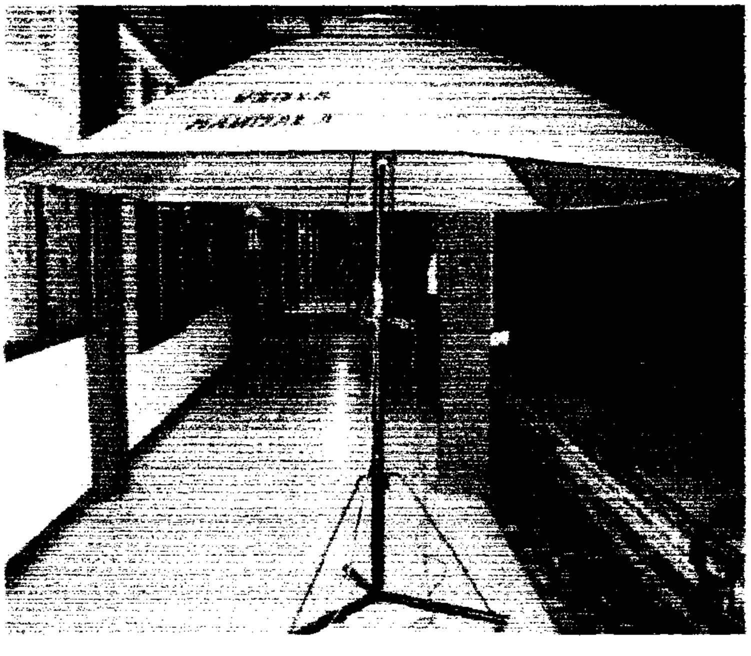
Gambar Lengkap Rangkaian



DAFTAR GAMBAR



Gambar payung saat tertutup



Gambar Payung saat terbuka

```
;==============================================================
;       PERACANGAN DAN IMPLEMENTASI PEMBUKA DAN PENUTUP
;                    PAYUNG BESAR SECARA ELEKTRONIK
;                          SUNOTO PARDI / 5103002023
;==============================================================

Port_Output     EQU     P2
RS_LCD          EQU     P3.0
En_LCD          EQU     P3.1
satu                    equ p1.0
dua                     equ p1.1
tiga                    equ p1.2
empat                   equ p1.3
ldr1                    equ p1.4
ldr2                    equ p1.5
relay1                  equ p1.6
relay2                  equ p1.7
LS_atas                  equ p3.2
LS_bawah                equ p3.3

bank                    data            18h
buffer                  data            19H
Delay_1                 DATA            20H
Delay_2                 DATA            21H
Delay_3                 DATA            22H

org  0000h
jmp 0100h
org 0100h


CALL            INIT_LCD
                        MOV             DPTR,#judul1
                        CALL            LCD_LINE_1
                        CALL            LCD
                        MOV             DPTR,#judul2
                        CALL            LCD_LINE_2
                        CALL            LCD

call delay_1s
pertama :
        setb relay1
        setb relay2
        call init_lcd
        mov dptr,#menu1
CALL    LCD_LINE_1
                        CALL            LCD
                        MOV             DPTR,#menu2
                        CALL            LCD_LINE_2
                        CALL            LCD

menu :
jb satu,cek_lagi
jmp menu_satu
cek_lagi :
```

```
jb dua,cek_lagi2
jmp menu_dua
cek_lagi2 :
jb tiga,cek_lagi3
jmp menu_tiga
cek_lagi3 :
jb empat,menu
ljmp setting_time_date
;==========================================================
;Program alamat baris di LCD
;==========================================================
LCD_LINE_1    :
              MOV       Port_Output,#80H        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_1:
              MOV       Port_Output,#84H        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_2:
              MOV       Port_Output,#85H        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_3:
              MOV       Port_Output,#87H        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_4:
              MOV       Port_Output,#88H        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_5:
              MOV       Port_Output,#8aH        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_1_6:
              MOV       Port_Output,#8bH        ; GotoXy (1,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              RET
LCD_LINE_2    :
              MOV       Port_Output,#0C0H        ; GotoXy (2,1)
              CALL      SENT_INIT
              CALL      Delay100uS
              ret
LCD_LINE_2_4:
              MOV       Port_Output,#0c8H        ; GotoXy (2,4)
              CALL      SENT_INIT
              CALL      Delay100uS
```

```
                RET

LCD_LINE_2_5:
                MOV             Port_Output,#0C4H           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_10       :
                MOV             Port_Output,#0C7H           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_11       :
                MOV             Port_Output,#0C8H           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_9:
                MOV             Port_Output,#0C9H           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_20       :
                MOV             Port_Output,#0CaH           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_12       :
                MOV             Port_Output,#0CcH           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_13       :
                MOV             Port_Output,#0CdH           ; GotoXy (2,1)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
LCD_LINE_2_15       :
                MOV             Port_Output,#0C7H           ; GotoXy (2,15)
                CALL            SENT_INIT
                CALL            Delay100uS
                ret
SENT  :
                SETB            RS_LCD
                SETB            En_LCD
                CLR     En_LCD
                CALL            Delay100uS
                RET
;===========================================================================
;                               Program MENU SATU               ;
;===========================================================================
menu_satu :
                call reset
                setb satu
                setb dua
                setb tiga
                setb empat
CALL    INIT_LCD
in:
        jnb ldr1,cek_ldr2
        jmp cek_ldr2

cek_ldr2 :
        jnb ldr2,gelap
        jmp terang

gelap :
                MOV             DPTR,#aa
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#aaaaa
                CALL            LCD_LINE_2
                CALL            LCD
                call delay100ms
                jb empat,next
                jmp pertama
next :
                setb relay1
                clr relay2
                jnb LS_bawah,gelap
                setb relay1
                setb relay2
                jb empat,ulang
                jmp pertama
ulang :
                jnb ldr1,more
                jb empat,more
                jmp pertama
more :
                jb empat,aduh
                jmp pertama
aduh :
                jnb ldr2 ,ulang
                jmp terang
terang :
                jb empat,hhh
jmp pertama
hhh:            MOV             DPTR,#aaa
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#aaaa
                CALL            LCD_LINE_2
                CALL            LCD
                jb empat,next1
                jmp pertama
next1 :         clr relay1
                clr relay2
                jnb LS_atas,terang
                setb relay1
                setb relay2
                jb empat,lola
                jmp pertama
lola :
                jb ldr1,wow
lolali :
                jb ldr2,wow1
                jb empat,rully
                jmp pertama
rully :
                jmp gelap

wow :
                jb empat,lola
                jmp pertama
wow1 :
                jb empat,lolali
                jmp pertama
                jmp $
;================================================================
                                   ;Program MENU dua      ;
;================================================================
menu_dua :
;call reset
        setb satu
        setb dua
        setb tiga
        setb empat
        setb relay1
        setb relay2
        mov r7,#00h
        mov r2,#00h
        clr LS_atas
        clr LS_bawah

; mengecek kondisi payung sebelumnya
cek_cok :
        jb empat,nobita
        jmp pertama
nobita :
        jnb LS_atas,cek_tutup
        mov r2,#00h
        jmp kageciyo
cek_tutup :
        jnb LS_bawah,cek_cok
        mov r2,#0ffh
        jmp hatori
kageciyo :
                call hapus_layar
                MOV             DPTR,#n2
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#n
                CALL            LCD_LINE_2
                CALL            LCD
                jmp             doraemon
```

```
hatori :
                    call hapus_layar
                    MOV             DPTR,#n
                    CALL            LCD_LINE_1
                    CALL            LCD
                    MOV             DPTR,#n2
                    CALL            LCD_LINE_2
                    CALL            LCD
                    jmp             doraemon
doraemon :
                    mov dptr,#v1
                    call lcd_line_2_15
                    call lcd
                    mov dptr,#v1
                    call lcd_line_1_3
                    call lcd
                    mov bank,#00H
                    mov dptr,#v1
                    call lcd_line_2_15
                    call lcd
masukkan :
                    mov buffer,#01H
                    mov dptr,#v1
                    call lcd_line_1_3
                    call lcd
masuk :
                    mov r6,#00h
tunggu_1 :
                    call delay100ms
                    jb empat,lanjut_bro1
                    jmp pertama
lanjut_bro1 :
                    call delay_200ms
                    jnb satu,tambah1
                    jnb tiga,kurang1
                    jnb dua,njero
                    jmp tunggu_1
njero :
                    call delay_200ms
                    ljmp enter
tambah1 :
                    inc r6
                    jmp proses1
kurang1 :
                    cjne r6,#00h,uhui1
                    mov r6,#18h
                    uhui1 : dec r6
proses1 :
                    cjne r6,#01h,uji_0_1
                    mov buffer,#02H
                    mov dptr,#v2
                    call lcd_line_1_3
                    call lcd
                    jmp tunggu_1
```

```
uji_0_1 :
                cjne r6,#02h,uji_1_1
                mov buffer,#03H
                mov dptr,#v3
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_1_1 :
                cjne r6,#03h,uji_2_1
                mov buffer,#04H
                mov dptr,#v4
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_2_1 :
                cjne r6,#04h,uji_3_1
                mov buffer,#05H
                mov dptr,#v5
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_3_1:
                cjne r6,#05h,uji_4_1
                mov buffer,#06H
                mov dptr,#v6
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_4_1 :
                cjne r6,#06h,uji_5_1
                mov buffer,#07H
                mov dptr,#v7
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_5_1 :
                cjne r6,#07h,uji_6_1
                mov buffer,#08H
                mov dptr,#v8
                call lcd_line_1_3
                call lcd
                jmp tunggu_1

uji_6_1 :
                cjne r6,#08h,uji_7_1
                mov buffer,#09H
                mov dptr,#v9
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_7_1 :
                cjne r6,#09h,uji_8_1
                mov buffer,#10H
                mov dptr,#v10
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_8_1:
                cjne r6,#0ah,uji_9_1
                mov buffer,#11H
                mov dptr,#v11
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_9_1 :
                cjne r6,#0bh,uji_10_1
                mov buffer,#12H
                mov dptr,#v12
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_10_1:
                cjne r6,#0ch,uji_11_1
                mov buffer,#13H
                mov dptr,#v13
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_11_1:
                cjne r6,#0dh,uji_12_1
                mov buffer,#14H
                mov dptr,#v14
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_12_1:
                cjne r6,#0eh,uji_13_1
                mov buffer,#15H
                mov dptr,#v15
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_13_1:
                cjne r6,#0fh,uji_14_1
                mov buffer,#16H
                mov dptr,#v16
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_14_1:
                cjne r6,#10h,uji_15_1
                mov buffer,#17H
                mov dptr,#v17
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_15_1:
                cjne r6,#11h,uji_16_1
                mov buffer,#18H
                mov dptr,#v18
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_16_1:
                cjne r6,#12h,uji_17_1
                mov buffer,#19H
                mov dptr,#v19
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_17_1:
                cjne r6,#13h,uji_18_1
                mov buffer,#20H
                mov dptr,#v20
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_18_1:
                cjne r6,#14h,uji_19_1
                mov buffer,#21H
                mov dptr,#v21
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_19_1:
                cjne r6,#15h,uji_20_1
                mov buffer,#22H
                mov dptr,#v22
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_20_1:
                cjne r6,#16h,uji_21_1
                mov buffer,#23H
                mov dptr,#v23
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_21_1:
                cjne r6,#17h,uji_22_1
                mov buffer,#00H
                mov dptr,#v24
                call lcd_line_1_3
                call lcd
                jmp tunggu_1
uji_22_1 :
                ljmp masukkan

;==========================================================
enter :
                mov bank,#01H
                mov dptr,#v1
                call lcd_line_2_15
                call lcd
                mov r7,#00h
tunggu :
                call delay100ms
                jb empat,lanjut_bro
                jmp pertama
lanjut_bro :
                call delay_200ms
                jnb satu,tambah
                jnb tiga,kurang
                jnb dua,mlebu
                jmp tunggu
mlebu :
                call delay_200ms
                ljmp tampil
tambah :
                inc r7
                jmp proses
kurang :
                cjne r7,#00h,uhui
                mov r7,#18h
                uhui : dec r7
                ;jmp proses
proses :
                cjne r7,#01h,uji_0
                mov bank,#02h
                mov dptr,#v2
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_0 :
                cjne r7,#02h,uji_1
                mov bank,#03h
                mov dptr,#v3
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_1 :
                cjne r7,#03h,uji_2
                mov bank,#04h
                mov dptr,#v4
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_2 :
                cjne r7,#04h,uji_3
                mov bank,#05h
                mov dptr,#v5
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_3:
                cjne r7,#05h,uji_4
                mov bank,#06h
                mov dptr,#v6
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_4 :
                cjne r7,#06h,uji_5
                mov bank,#07h
                mov dptr,#v7
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_5 :
                cjne r7,#07h,uji_6
                mov bank,#08h
                mov dptr,#v8
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_6 :
                cjne r7,#08h,uji_7
                mov bank,#09h
                mov dptr,#v9
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_7 :
                cjne r7,#09h,uji_8
                mov bank,#10h
                mov dptr,#v10
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_8 :
                cjne r7,#0ah,uji_9
                mov bank,#11h
                mov dptr,#v11
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_9 :
                cjne r7,#0bh,uji_10
                mov bank,#12h
                mov dptr,#v12
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_10:
                cjne r7,#0ch,uji_11
                mov bank,#13h
                mov dptr,#v13
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_11:
                cjne r7,#0dh,uji_12
                mov bank,#14h
                mov dptr,#v14
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_12:
                cjne r7,#0eh,uji_13
                mov bank,#15h
                mov dptr,#v15
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_13:
                cjne r7,#0fh,uji_14
                mov bank,#16h
                mov dptr,#v16
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_14:
                cjne r7,#10h,uji_15
                mov bank,#17h
                mov dptr,#v17
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_15:
                cjne r7,#11h,uji_16
                mov bank,#18h
                mov dptr,#v18
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_16:
                cjne r7,#12h,uji_17
                mov bank,#19h
                mov dptr,#v19
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_17:
                cjne r7,#13h,uji_18
                mov bank,#20h
                mov dptr,#v20
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_18:
                cjne r7,#14h,uji_19
                mov bank,#21h
                mov dptr,#v21
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_19:
                cjne r7,#15h,uji_20
                mov bank,#22h
                mov dptr,#v22
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_20:
                cjne r7,#16h,uji_21
                mov bank,#23h
                mov dptr,#v23
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_21:
                cjne r7,#17h,uji_22
                mov bank,#24h
                mov dptr,#v24
                call lcd_line_2_15
                call lcd
                jmp tunggu
uji_22:
                jmp enter

tampil :
                mov 23H,buffer
                mov 24H,bank
                setb satu
                setb dua
                setb tiga
                setb empat
                setb relay1
                setb relay2
suneo :
                lcall delay_200ms
;CALL init_lcd
                CALL hapus_layar

MOV             DTR,#jam
                CALL            LCD_LINE_1
                CALL            LCD
MOV             DPTR,#laluna
                CALL            LCD_LINE_2
                CALL            LCD
MOV     A,23H
                MOV B,A
                SWAP A
                ANL A,#0FH
                RL A,#30H
                MOV             Port_Output,#80H
                CALL            SENT_INIT
                CALL            Delay100Us
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                OV      Port_Output,#81h
                CALL            SENT_INIT
                CALL            Delay100uS
                MOV PORT_OUTPUT,A
                CALL SENT
MOV     A,24H
                MOV B,A
                WAP A
                ANL A,#0FH
                ORL A,#30H
                MOV             Port_Output,#82H
                CALL            SENT_INIT
                CALL            Delay100uS
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                RL A,#30H
                MOV             Port_Output,#83h
                CALL            SENT_INIT
                CALL            Delay100uS
                MOV PORT_OUTPUT,A
                CALL SENT
OV1 :
                MOV R0,#4
                MOVX A,@R0
                mov R5,A
                MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_1
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                 ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_2
                MOV PORT_OUTPUT,A
                CALL SENT

ambil_menit :
MOV R0,#2       ;point to minute loc
MOVX  A,@R0 ;read minute
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_3
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_4
                MOV PORT_OUTPUT,A
                CALL SENT
MOV R0,#0               ;point to second 1oc
MOVX A,@R0  ;read second
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_5
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_6
                MOV PORT_OUTPUT,A
                CALL SENT
pat_kay :
                jnb empat,jauh
                ajmp    lompat
jauh :
                call delay_200ms
                ljmp menu_dua
lompat:
                cjne R2,#00h,sarah_azhari  ; payung siap ditutup
                mov A,R5
                cjne A,23H,albert_gunadhi
                jmp tutup_payung_dulu
albert_gunadhi :
                cjne A,24H,rahma_azhari
                jmp buka_payung_dulu
sarah_azhari :                          ; payung siap dibuka
                ajmp buka_baru_tutup
rahma_azhari :
                LJMP ov1                ; display date forever
buka_baru_tutup :
                  Mov A,R5
                cjne A,23H,albert_gunadhi_ST
                jmp BUKA_payung_dulu
albert_gunadhi_ST :
                cjne A,24H,rahma_azhari
                jmp TUTUP_payung_dulu
buka_payung_dulu :
                jb LS_atas,motor_mati2
                ajmp    boleh_buka
buka_payung_dulu_2 :
                jb LS_ATAS,motor_mati2
                ajmp    boleh_buka
motor_mati2:
                setb relay1
                setb relay2
                ajmp    tampil
boleh_buka:
```

```
                clr relay1
                clr relay2
lcall delay_200ms
;CALL hapus_layar
MOV             DPTR,#jam
                CALL            LCD_LINE_1
                CALL            LCD
MOV             DPTR,#laluna
                CALL            LCD_LINE_2
                CALL            LCD
MOV R0,#4
                MOVX A,@R0
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_1
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_2
                MOV PORT_OUTPUT,A
                CALL SENT
;jmp ambil_menit
                MOV R0,#2               ;point to minute loc
                MOVX  A,@R0  ;read minute
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_3
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_4
                MOV PORT_OUTPUT,A
                CALL SENT
MOV R0,#0                    ;point to second 1oc
MOVX A,@R0  ;read second
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_5
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_6
                MOV PORT_OUTPUT,A
                CALL SENT
ljmp buka_payung_dulu
ljmp tampil
TUTUP_payung_dulu_2 :
                  jb      LS_bawah,motor_mati1
                ajmp    boleh_tutup
tutup_payung_dulu :

                jb      LS_bawah,motor_mati1
                ajmp    boleh_tutup
motor_mati1:
                setb relay1
                setb relay2
                ajmp    tampil
boleh_tutup:
                setb relay1
                clr relay2
lcall delay_200ms
;CALL hapus_layar
MOV             DPTR,#jam
                CALL            LCD_LINE_1
                CALL            LCD
MOV             DPTR,#laluna
                CALL            LCD_LINE_2
                CALL            LCD
MOV R0,#4
                MOVX A,@R0
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_1
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_2
                MOV PORT_OUTPUT,A
                CALL SENT
;jmp ambil_menit
                MOV R0,#2               ;point to minute loc
                MOVX  A,@R0  ;read minute
MOV B,A
                SWAP A
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_3
                MOV PORT_OUTPUT,A
                CALL SENT
                MOV A,B
                ANL A,#0FH
                ORL A,#30H
                CALL LCD_LINE_1_4
                MOV PORT_OUTPUT,A
                CALL SENT

MOV R0,#0               ;point to second 1oc
MOVX A,@R0      ;read second
MOV B,A
                        SWAP A
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_5
                        MOV PORT_OUTPUT,A
                        CALL SENT
                        MOV A,B
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_6
                        MOV PORT_OUTPUT,A
                        CALL SENT
ljmp tutup_payung_dulu
ljmp tampil
teruskan_jam :
                        lcall delay_200ms
                        CALL hapus_layar
MOV             DPTR,#jam
                        CALL            LCD_LINE_1
                        CALL            LCD
MOV             DPTR,#laluna
                        CALL            LCD_LINE_2
                        CALL            LCD
MOV R0,#4
                        MOVX A,@R0
MOV B,A
                        SWAP A
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_1
                        MOV PORT_OUTPUT,A
                        CALL SENT
                        MOV A,B
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_2
                        MOV PORT_OUTPUT,A
                        CALL SENT
;jmp ambil_menit
MOV R0,#2               ;point to minute loc
MOVX  A,@R0             ;read minute
MOV B,A
                        SWAP A
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_3
                        MOV PORT_OUTPUT,A
                        CALL SENT
                        MOV A,B
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_4
                        MOV PORT_OUTPUT,A
                        CALL SENT
MOV R0,#0               ;point to second 1oc
MOVX A,@R0              ;read second
MOV B,A
                        SWAP A
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_5
                        MOV PORT_OUTPUT,A
                        CALL SENT
                        MOV A,B
                        ANL A,#0FH
                        ORL A,#30H
                        CALL LCD_LINE_1_6
                        MOV PORT_OUTPUT,A
                        CALL SENT
jmp tutup_payung_dulu
DELAY :
                        MOV R7,#250
D1:                     DJNZ R7, D1
                        RET
;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                      ;  procedure untuk mensetting jam dan tanggal
;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Setting_time_date :     mov  dptr,#clock1
                        call lcd_line_1
                        call lcd
                        mov  dptr,#clock2
                        call lcd_line_2
                        call lcd
angka_jam :             mov r4,#00H
                        mov dptr,#v24
                        ljmp kiriman
                        mov r3,#00h
                        mov dptr,#t60
                        ljmp kiriman1
wait_jam :
                        call delay_200ms
                        jnb satu,aduk_jam
                        jnb tiga,aduk_menit
                        jb dua,wait_jam
                        jmp kirim_rtc
aduk_jam :
                        inc r4
                        ljmp adonan
aduk_menit :
                        INC R3
azhari :
                        cjne r3,#00h,mama0
                        mov dptr,#t60
                        ljmp kiriman1
mama0 :                 cjne r3,#01h,mama1
```

```
                    mov r3,#05h
                    mov dptr,#t5
                    ljmp kiriman1
mama1 :
                    cjne r3,#06h,mama2
                    mov r3,#10h
                    mov dptr,#t10
                    ljmp kiriman1
mama2 :
                    cjne r3,#11h,mama3
                    mov  r3,#15h
                    mov dptr,#t15
                    ljmp kiriman1
mama3 :
                    cjne r3,#16h,mama4
                    mov r3,#20h
                    mov dptr,#t20
                    ljmp kiriman1
mama4 :
                    cjne r3,#21h,mama5
                    mov r3,#25h
                    mov dptr,#t25
                    ljmp kiriman1
mama5 :
                    cjne r3,#26h,mama6
                    mov r3,#30h
                    mov dptr,#t30
                    ljmp kiriman1
mama6 :
                    cjne r3,#31h,mama7
                    mov r3,#35h
                    mov dptr,#t35
                    ljmp kiriman1
mama7 :
                    cjne r3,#36h,mama8
                    mov r3,#40h
                    mov dptr,#t40
                    ljmp kiriman1
mama8 :
                    cjne r3,#41h,mama9
                    mov r3,#45h
                    mov dptr,#t45
                    ljmp kiriman1
mama9 :
                    cjne r3,#46h,mama10
                    mov  r3,#50h
                    mov dptr,#t50
                    ljmp kiriman1
mama10 :
                    cjne r3,#51h,mama11
                    mov r3,#55h
                    mov dptr,#t55
                    ljmp kiriman1
mama11 :
                    cjne r3,#56h,mama12
```

```
                              mov r3,#60h
                              mov dptr,#t60
                              ljmp kiriman1
mama12 :  cjne r3,#61h,xxx
                              mov r3,#00h
xxx: ljmp wait_jam
kiriman1 :
                    call lcd_line_2_4
                    call lcd
                    jmp wait_jam
adonan :
                    cjne r4,#01h,jam1
                    mov dptr,#v1
                    ljmp kiriman
jam1 :
                    cjne r4,#02h,jam2
                    mov dptr,#v2
                    ljmp kiriman
jam2 :
                    cjne r4,#03h,jam3
                    mov dptr,#v3
                    ljmp kiriman
jam3 :
                    cjne r4,#04h,jam4
                    mov dptr,#v4
                    ljmp kiriman
jam4 :
                    cjne r4,#05h,jam5
                    mov dptr,#v5
                    ljmp kiriman
jam5 :
                    cjne r4,#06h,jam6
                    mov dptr,#v6
                    ljmp kiriman
jam6 :
                    cjne r4,#07h,jam7
                    mov dptr,#v7
                    ljmp kiriman
jam7 :
                    cjne r4,#08h,jam8
                    mov dptr,#v8
                    ljmp kiriman

jam8 :
                    cjne r4,#09h,jam9
                    mov dptr,#v9
                    ljmp kiriman
jam9 :
                    cjne r4,#0ah,jam10
                    mov r4,#10h
                    mov dptr,#v10
                    ljmp kiriman
jam10:
                    cjne r4,#11h,jam11
                    mov dptr,#v11
                    ljmp kiriman
jam11:
                    cjne r4,#12h,jam12
                    mov dptr,#v12
                    ljmp kiriman
jam12:
                    cjne r4,#13h,jam13
                    mov dptr,#v13
                    ljmp kiriman
jam13:
                    cjne r4,#14h,jam14
                    mov dptr,#v14
                    ljmp kiriman
jam14:
                    cjne r4,#15h,jam15
                    mov dptr,#v15
                    ljmp kiriman
jam15:
                    cjne r4,#16h,jam16
                    mov dptr,#v16
                    ljmp kiriman
jam16:
                    cjne r4,#17h,jam17
                    mov dptr,#v17
                    ljmp kiriman
jam17:
                    cjne r4,#18h,jam18
                    mov dptr,#v18
                    ljmp kiriman
jam18:
                    cjne r4,#19h,jam19
                    mov dptr,#v19
                    ljmp kiriman
jam19:
                    cjne r4,#1ah,jam20
                    mov r4,#20h
                    mov dptr,#v20
                    ljmp kiriman
jam20:
                    cjne r4,#21h,jam21
                    mov dptr,#v21
                    ljmp kiriman
jam21:
                    cjne r4,#22h,jam22
                    mov dptr,#v22
                    ljmp kiriman
jam22:
                    cjne r4,#23h,jam23
                    mov dptr,#v23
                    ljmp kiriman
jam23:
                    cjne r4,#24h,jam24
                    mov dptr,#v24
                    ljmp kiriman
jam24 :
```

```
                 ljmp angka_jam
kiriman :
                call lcd_line_1_4
                call lcd
                jmp wait_jam
kirim_rtc :
                lcall delay_200ms
                MOV R0,#10
                MOV A,#20H
                MOVX @R0,A
                MOV R0,#11
                MOV A,#83H
                MOVX @R0,A
                MOV R0,#0                       ; point to second addres
                MOV A,#01H              ; second=55h (BCD numbers need H)
                MOVX @R0,A              ; set second
                MOV R0,#02                      ; point to minutes addres
                MOV A,r3                ; minutes= 58
                MOVX @R0,A              ; set minutes
                MOV R0,#04                      ; point to hours addres
                MOV A,r4                ; second=16
                MOVX @R0,A              ; set hours
                MOV R0,#11                      ; Reg B addres
                MOV A,#03                       ; D7=0 of reg B to allow update
                MOVX @R0,A              ; send it to reg B
;------TURNING ON THE RTC
                MOV R0,#10                      ; R0=0AH, Reg A addres
                MOV A,#20H              ;  010 IN D6-D4 turn on osc
                MOVX @R0,A              ; send it to Reg A of DS12887
;------SETTING THE TIME MODE
                MOV R0,#11                      ; Reg B addres
                MOV A,#83H              ; BCD, 24 hrs, Daylight saving, D7=1 No update
                MOVX @R0,A              ; send it to Reg B
                MOV R0,#07                      ; load pointer for DAY OF MONTH
                MOV A,#17H                      ; DAY=17h (BCD  numbers need H)
                MOVX @R0,A              ; set DAY OF MONTH
                lCALL DELAY             ;
                MOV R0,#08                      ; point to MONTH
                MOV A,#02H              ; 02=FEBRUARY
                MOVX @R0,A              ; set MONTH
                lCALL DELAY
                MOV R0,#09                      ; point to YEAR addres
                MOV A,#07                       ; YEAR=07 FOR 2007
                MOVX @R0,A              ; set YEAR to 2007
                lCALL DELAY
                MOV R0,#11                      ; Reg B addres
                MOV A,#03                       ; D7=0 of reg B to allow update
                MOVX @R0,A              ; send it to reg B
call delay_200ms
jb dua,$
ljmp pertama
;==========================================================================
                                          ;Program MENU tiga;
;==========================================================================
menu_tiga :
```

```
                call reset
                setb satu
                setb dua
                setb tiga
                setb empat
CALL            INIT_LCD
manual :
                MOV             DPTR,#m
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#m1
                CALL            LCD_LINE_2
                CALL            LCD
analisa :
                jb satu,cek_dua
terus :         jb empat,lucy  ;baru
                jmp pertama                ;baru
lucy:           clr relay1
                clr relay2
                jnb LS_atas,cek_stop
                setb relay1
                setb relay2
ngatini :
                jb empat,katrok
                jmp pertama
katrok :
                jb  tiga,ngatini
                jmp terus_1
cek_stop :
                jb dua,terus
                setb relay1
                setb relay2
                MOV             DPTR,#m
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#m1
                CALL            LCD_LINE_2
                CALL            LCD
                jb empat,tukul
                jmp pertama
tukul :
                jmp analisa
cek_dua         :
                jb tiga,cek_back
terus_1 :
                jb empat,laila   ;baru
                jmp pertama                ;baru
laila :
                setb relay1
                clr relay2
                jnb LS_bawah,cek_stop_2
                setb relay1
                setb relay2
ngatiyem :
                jb empat,ndeso
                jmp pertama
ndeso :
                jb satu,ngatiyem
                jmp terus

cek_stop_2 :
                jb dua,terus_1
                setb relay1
                setb relay2
                MOV             DPTR,#m
                CALL            LCD_LINE_1
                CALL            LCD
                MOV             DPTR,#m1
                CALL            LCD_LINE_2
                CALL            LCD
                jb empat,tukul_arwana
                jmp pertama
tukul_arwana :
                jmp analisa
cek_back :
                jb empat,culun
                jmp pertama
culun :
                 jmp manual
jmp $
;=====================================================================
                ;Program Inisialisasi LCD                                  ;
;=====================================================================
INIT_LCD                :
                CALL            Delay100mS
                MOV             Port_Output,#30H        ; Function set 8 bit
                CALL            SENT_INIT
                CALL    Delay20mS
                MOV             Port_Output,#30H        ; Function set 8 bit
                CALL            SENT_INIT
                CALL    Delay20mS
                MOV             Port_Output,#30H        ; Function set 8 bit
                CALL            SENT_INIT
                CALL   Delay20mS
                MOV             Port_Output,#3cH
                CALL            SENT_INIT
                CALL            Delay100uS
                MOV             Port_Output,#0eH        ;baru dgnt
                ALL             SENT_INIT
                CALL            Delay100uS
                MOV             Port_Output,#01H        ; Clear display
                CALL            SENT_INIT
                CALL            Delay2mS
                MOV             Port_Output,#07H        ;entry mode br dgnt 07
                CALL            SENT_INIT
                CALL            Delay100uS
                MOV             Port_Output,#06H        ; Cursor INCrement,
                CALL            SENT_INIT
                CALL            Delay100uS
```

```
                    MOV         Port_Output,#00001100b        ; Display on,
cursor off,
                    CALL        SENT_INIT               ; blink off
                    CALL        Delay100uS
                    RET
;=====================================================================
                 ;Program Clear Display                  ;
;=====================================================================
HAPUS_LAYAR                 :
                    MOV         Port_Output,#01H        ; Clear display
                    CALL        SENT_INIT
                    CALL        Delay2mS
                    RET
SENT_INIT           :
                    CLR    RS_LCD
                    SETB        En_LCD
                    CLR    En_LCD
                    RET
;=====================================================================
                        ;Program Reset                ;
;=====================================================================
RESET          :
                    MOV         P2,#0ffh
                    mov    p1,#11000000b
                    MOV         P0,#00h
                    MOV         P3,#00h
                    MOV         A,#00h
                    mov         r0,#00h
                    MOV         R1,#00h
                    MOV         R2,#00h
                    MOV         R3,#00h
                    MOV         R4,#00h
                    MOV         R5,#00h
                    MOV         R6,#00h
                    MOV         R7,#00h
                    RET
;=====================================================================
                 ;Program menampilkan pada LCD
;=====================================================================
LCD            :
                    CLR         A
                    MOVC        A,@A+DPTR
                    CJNE        A,#0h,CEK
                    RET
CEK            :
                    MOV         Port_Output,A
                    CALL        SENT
                    INC         DPTR
                    JMP         LCD
;==============================================================
;Program delay
;==============================================================
Delay100uS     :
                    MOV         Delay_1,#50
Loop_Delay100uS :
```

```
                     DJNZ     Delay_1,$
                     RET
Delay2mS        :
                     MOV           Delay_2,#4
Loop_Delay2mS        :
                     MOV           Delay_1,#250
                     DJNZ          Delay_1,$
                     DJNZ          Delay_2,Loop_Delay2mS
                     RET
Delay20mS       :
                     MOV           Delay_2,#40
Loop_Delay20mS       :
                     MOV           Delay_1,#250
                     DJNZ          Delay_1,$
                     DJNZ          Delay_2,Loop_Delay20mS
                     RET
Delay100mS:
                     MOV           Delay_2,#200
Loop_Delay100mS:
                     MOV           Delay_1,#250
                     DJNZ          Delay_1,$
                     DJNZ          Delay_2,Loop_Delay100mS
                     RET
delay500ms :
                     mov delay_3,#5
h :
                     call delay100ms
                     djnz delay_3,h
                     ret
delay_200ms :
                     mov delay_3,#2
here1           :
                     mov delay_2,#180
here2   :
                     mov delay_1,#255
here3   :
                     djnz     delay_1,here3
                     djnz     delay_2,here2
                     djnz     delay_3,here1
                     ret
DELAY_1S        :
                     MOV           Delay_3,#8
DELAY1          :
                     MOV           Delay_2,#255
DELAY2  :
                     MOV           Delay_1,#225
                     DJNZ          Delay_1,$
                     DJNZ          Delay_2,DELAY2
                     DJNZ          Delay_3,DELAY1
                     RET

awal :  db ' Harap Tunggu  ',0h
awal2 : db  ' Payung Menutup ',0h
judul1 : db 'Pembuka/Penutup ',0h
judul2 : db ' Payung Besar   ',0h
```

```
loading : db 'Loading . . . .',0h
menu1 : db  '1.Auto   2.Timer',0h
menu2 : db  '3.Manual 4.Atur ',0h
aa        : db    'Kondisi= Gelap ',0h
aaa       : db  'Kondisi= Terang',0h
aaaa : db   'Payung = Buka  ',0h
aaaaa : db  'Payung = Tutup ',0h
clock1 : db 'Jam   =       ',0h
clock2 : db 'Menit = 00    ',0h
m : db                'Manual Operation',0h
m1 : db    '1->Buka 3->Tutup',0h
n1 : db               '==Masukkan Jam==',0h
n : db                'Buka =>',0h
n2 : db               'Tutup=>',0h
alarm : db  'KRIIIIIIIIIIIING',0H
jam : db    '====__:__:__====',0h
laluna : db 'Press 4 to back!',0h
tgl : db   '20',0h
v1 : db             '1 Pagi  ',0h
v2 :   db  '2 Pagi  ',0h
v3  :  db  '3 Pagi  ',0h
v4   : db  '4 Pagi  ',0h
v5    : db '5 Pagi  ',0h
v6    : db '6 Pagi  ',0h
v7   : db  '7 Pagi  ',0h
v8    : db '8 Pagi  ',0h
v9    : db '9 Pagi  ',0h
v10    : db '10 Pagi  ',0h
v11    : db '11 Pagi  ',0h
v12    : db        '12 Siang ',0h
v13   : db '1 Siang ',0h
v14    : db '2 Siang ',0h
v15    : db '3 Siang ',0h
v16    : db '4 Sore  ',0h
v17    : db '5 Sore  ',0h
v18     : db '6 Sore  ',0h
v19     : db '7 Malam ',0h
v20     : db '8 Malam ',0h
v21     : db '9 Malam ',0h
v22     : db '10 Malam',0h
v23     : db '11 Malam',0h
v24     : db '12 Malam',0h
t60 : db '00',0h
t: db '01',0h
t2          : db '02',0h
t3          : db '03',0h
t4          : db '04',0h
t5          : db '05',0h
t6          : db '06',0h
t7          : db '07',0h
t8          : db '08',0h
t9          : db '09',0h
t10         : db '10',0h
t11         : db '11',0h
t12         : db '12',0h
```

```
t13        : db '13',0h
t14        : db '14',0h
t15        : db '15',0h
t16        : db '16',0h
t17        : db '17',0h
t18        : db '18',0h
t19        : db '19',0h
t20        : db '20',0h
t21        : db '21',0h
t22        : db '22',0h
t23        : db '23',0h
t24        : db '24',0h
t25        : db '25',0h
t26        : db '26',0h
t27        : db '27',0h
t28        : db '28',0h
t29        : db '29',0h
t30        : db '30',0h
t31        : db '31',0h
t32        : db '32',0h
t33        : db '33',0h
t34        : db '34',0h
t35        : db '35',0h
t36        : db '36',0h
t37        : db '37',0h
t38        : db '38',0h
t39        : db '39',0h
t40        : db '40',0h
t41        : db '41',0h
t42        : db '42',0h
t43        : db '43',0h
t44        : db '44',0h
t45        : db '45',0h
t46        : db '46',0h
t47        : db '47',0h
t48        : db '48',0h
t49        : db '49',0h
t50        : db '50',0h
t51        : db '51',0h
t52        : db '52',0h
t53        : db '53',0h
t54        : db '54',0h
t55        : db '55',0h
t56        : db '56',0h
t57        : db '57',0h
t58        : db '58',0h
t59        : db '59',0h

end
```

Real-Time Clock

DS12885/DS12887/DS12887A/DS12C887/DS12C887A

General Description

ie DS12885, DS12887, and DS12C887 real-time ocks (RTCs) are designed to be direct replacements r the DS1285 and DS1287. The devices provide a al-time clock/calendar, one time-of-day alarm, three askable interrupts with a common interrupt output, a ogrammable square wave, and 114 bytes of battery-acked static RAM (113 bytes in the DS12C887 and S12C887A). The DS12887 integrates a quartz crystal d lithium energy source into a 24-pin encapsulated P package. The DS12C887 adds a century byte at ldress 32h. For all devices, the date at the end of the onth is automatically adjusted for months with fewer an 31 days, including correction for leap years. The evices also operate in either 24-hour or ?-hour format with an AM/PM indicator. A precision mperature-compensated circuit monitors the status of cc. If a primary power failure is detected, the device itomatically switches to a backup supply. A lithium in-cell battery can be connected to the V_{BAT} input n on the DS12885 to maintain time and date operation ien primary power is absent. The device is accessed rough a multiplexed byte-wide interface, which sup-rts both Intel and Motorola modes.

Applications

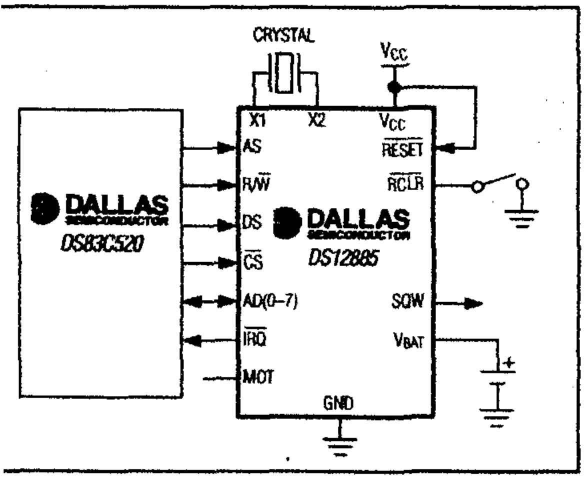
Embedded Systems

Utility Meters

Security Systems

Network Hubs, Bridges, and Routers

Typical Operating Circuit



Features

- ♦ Drop-In Replacement for IBM AT Computer Clock/Calendar
- ♦ RTC Counts Seconds, Minutes, Hours, Day, Date, Month, and Year with Leap Year Compensation Through 2099
- ♦ Binary or BCD Time Representation
- ♦ 12-Hour or 24-Hour Clock with AM and PM in 12-Hour Mode
- ♦ Daylight Saving Time Option
- ♦ Selectable Intel or Motorola Bus Timing
- ♦ Interfaced with Software as 128 RAM Locations
- ♦ 14 Bytes of Clock and Control Registers
- ♦ 114 Bytes of General-Purpose, Battery-Backed RAM (113 Bytes in the DS12C887 and DS12C887A)
- ♦ RAM Clear Function (DS12885, DS12887A, and DS12C887A)
- ♦ Interrupt Output with Three Independently Maskable Interrupt Flags
- ♦ Time-of-Day Alarm Once Per Second to Once Per Day
- ♦ Periodic Rates from 122µs to 500ms
- ♦ End-of-Clock Update Cycle Flag
- ♦ Programmable Square-Wave Output
- ♦ Automatic Power-Fail Detect and Switch Circuitry
- ♦ Optional 28-Pin PLCC Surface Mount Package or 32-Pin TQFP (DS12885)
- ♦ Optional Encapsulated DIP (EDIP) Package with Integrated Crystal and Battery (DS12887, DS12887A, DS12C887, DS12C887A)
- ♦ Optional Industrial Temperature Range Available
- ♦ Underwriters Laboratory (UL) Recognized

n Configurations and Ordering Information appear at end of data sheet.

or pricing, delivery, and ordering information, please contact Maxim/Dallas Direct! at 888-629-4642, or visit Maxim's website at www.maxim-ic.com.

Real-Time Clock

ABSOLUTE MAXIMUM RATINGS

Voltage Range on V_{CC} Pin Relative to Ground.....-0.3V to +6.0V
Operating Temperature Range ....
Commercial (noncondensing)..............................0°C to +70°C
Operating Temperature Range ....
Industrial (noncondensing)...............................-40°C to +85°C

Storage Temperature Range...............................-55°C to +125°C
Soldering Temperature.......................................See IPC/JEDEC
J-STD-020 Specification (Note 1)
Soldering Temperature (leads, 10s)..................................+260°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC ELECTRICAL CHARACTERISTICS

(**V_{CC} = +4.5V to +5.5V**, T_A = over the operating range, unless otherwise noted.) (Note 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Supply Voltage	V_{CC}	(Note 3)	4.5		5.5	V
V_{BAT} Input Voltage	V_{BAT}	(Note 3)	2.5		4.0	V
Input Logic 1	V_{IH}	(Note 3)	2.2		V_{CC} + 0.3	V
Input Logic 0	V_{IL}	(Note 3)	-0.3		+0.8	V
V_{CC} Power-Supply Current	I_{CC1}	(Note 4)			15	mA
V_{CC} Standby Current	I_{CCS}	(Note 5)				mA
Input Leakage	I_{IL}		-1.0		+1.0	µA
I/O Leakage	I_{OL}	(Note 6)	-1.0		+1.0	µA
Input Current	I_{MOT}	(Note 7)	-1.0		+500	µA
Output at 2.4V	I_{OH}	(Note 3)	-1.0			mA
Output at 0.4V	I_{OL}	(Note 3)			4.0	mA
Power-Fail Voltage	V_{PF}	(Note 3)	4.0	4.25	4.5	V
VRT Trip Point	VRT_{TRIP}			1.3		V

C ELECTRICAL CHARACTERISTICS

cc = 0V, V_{BAT} = 3.0V, T_A = over the operating range, unless otherwise noted.) (Note 2)

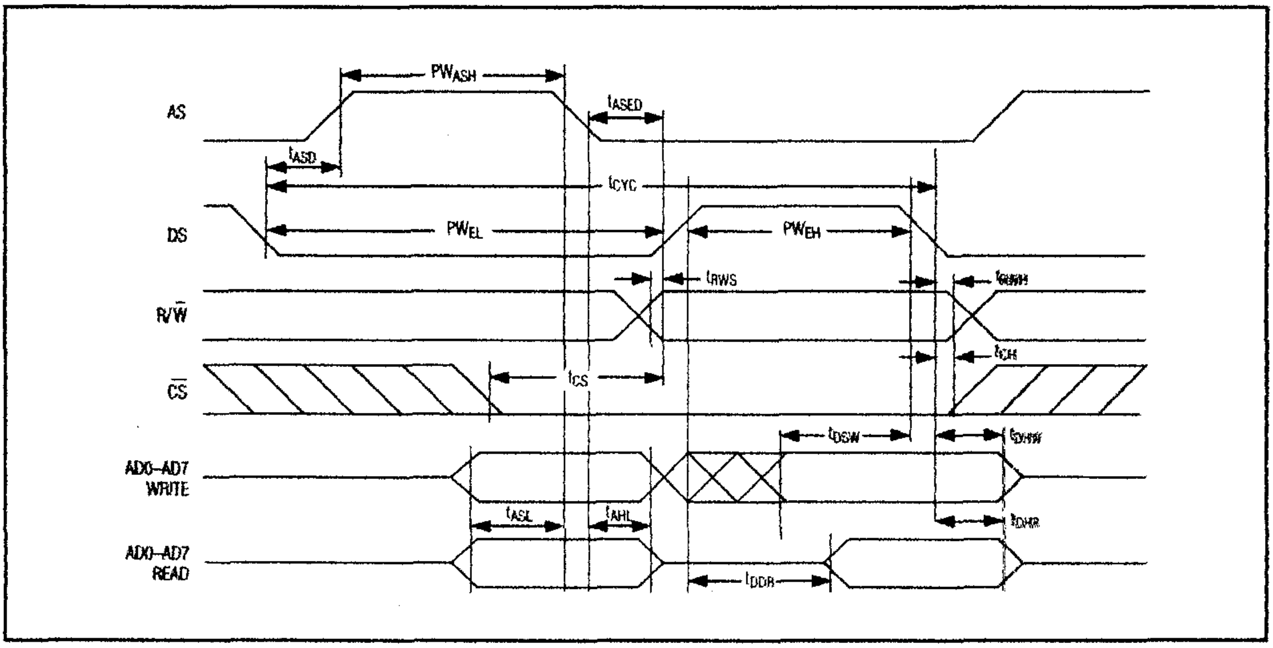
PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
BAT Current (OSC On); \ = +25°C, V_{BACKUP} = 3.0V	I_{BAT}	(Note 8)			500	nA
BAT Current (Oscillator Off)	I_{BATDR}	(Note 8)			100	nA

C ELECTRICAL CHARACTERISTICS

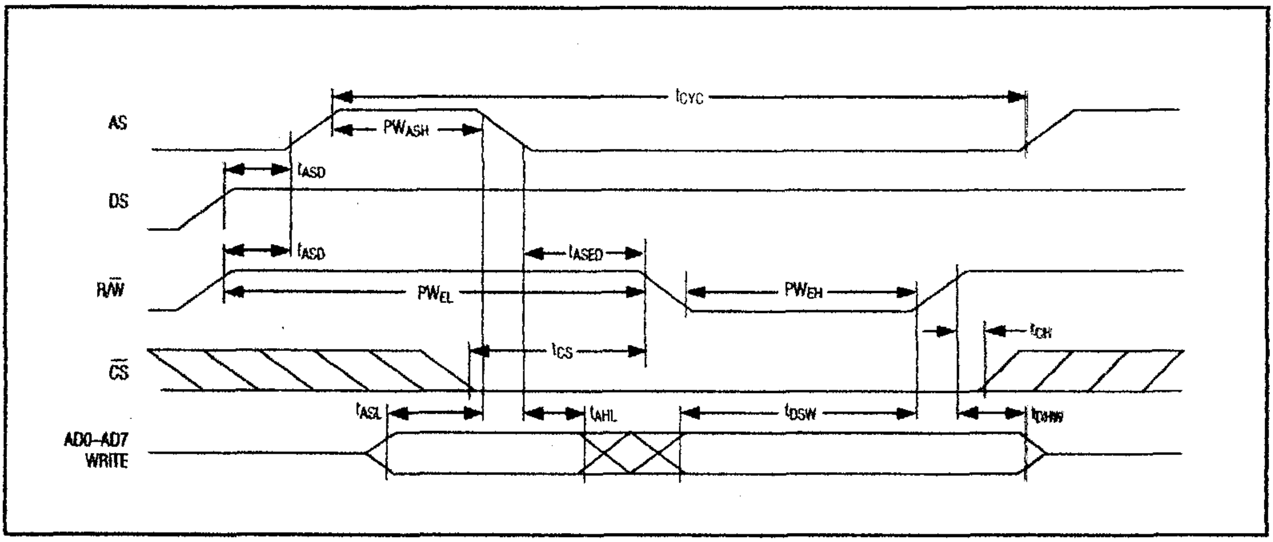
cc = 4.5V to 5.5V, T_A = over the operating range.) (Note 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
ycle Time	t_{CYC}		385		DC	ns
ılse Width, DS Low or R/$\overline{W}$ High	PW_{EL}		150			ns
ılse Width, DS High or R/$\overline{W}$ Low	PW_{EH}		125			ns
put Rise and Fall	t_R, t_F				30	ns
$\overline{W}$ Hold Time	t_{RWH}		10			ns
$\overline{W}$ Setup Time Before DS/E	t_{RWS}		50			ns
ıip-Select Setup Time Before S or R/$\overline{W}$	t_{CS}		20			ns
ıip-Select Hold Time	t_{CH}		0			ns
ead-Data Hold Time	t_{DHR}		10		80	ns
rite-Data Hold Time	t_{DHW}		0			ns
ldress Valid Time to AS Fall	t_{ASL}		30			ns
ldress Hold Time to AS Fall	t_{AHL}		10			ns
ılay Time DS/E to AS Rise	t_{ASD}		20			ns
ılse Width AS High	PW_{ASH}		60			ns
ılay Time, AS to DS/E Rise	t_{ASED}		40			ns
ıtput Data Delay Time from DS R/$\overline{W}$	t_{DDR}		20		120	ns
ıta Setup Time	t_{DSW}		100			ns
ıset Pulse Width	t_{RWL}		5			µs
$\overline{Q}$ Release from DS	t_{IRDS}				2	µs
$\overline{Q}$ Release from $\overline{RESET}$	t_{IRR}				2	µs

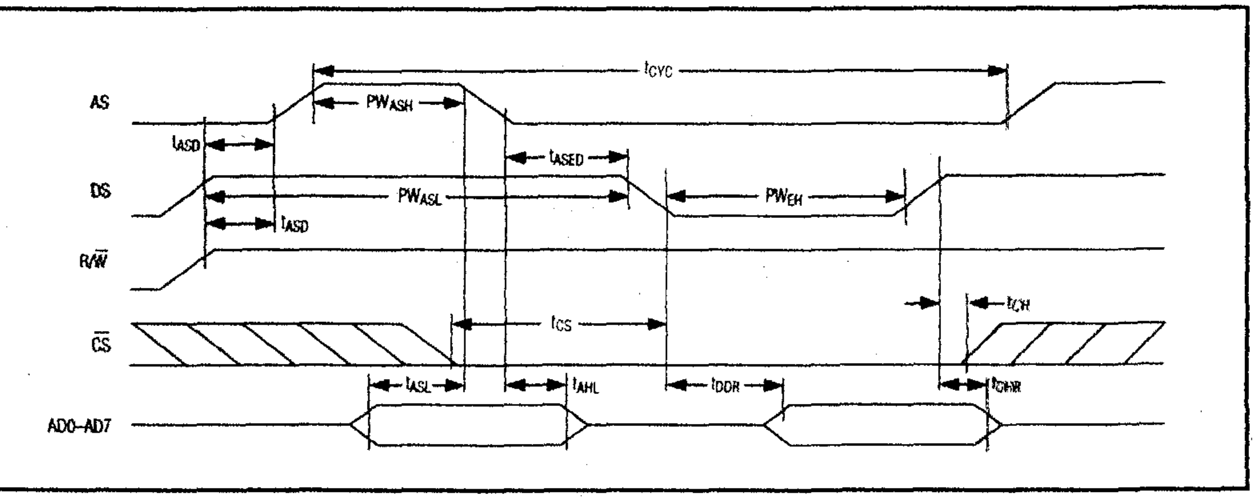
Motorola Bus Read/Write Timing



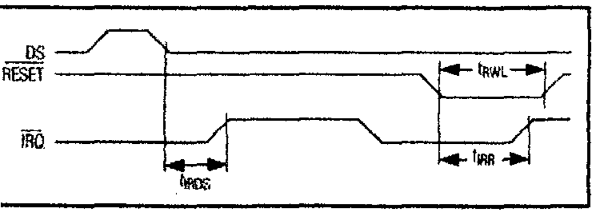
Intel Bus Write Timing



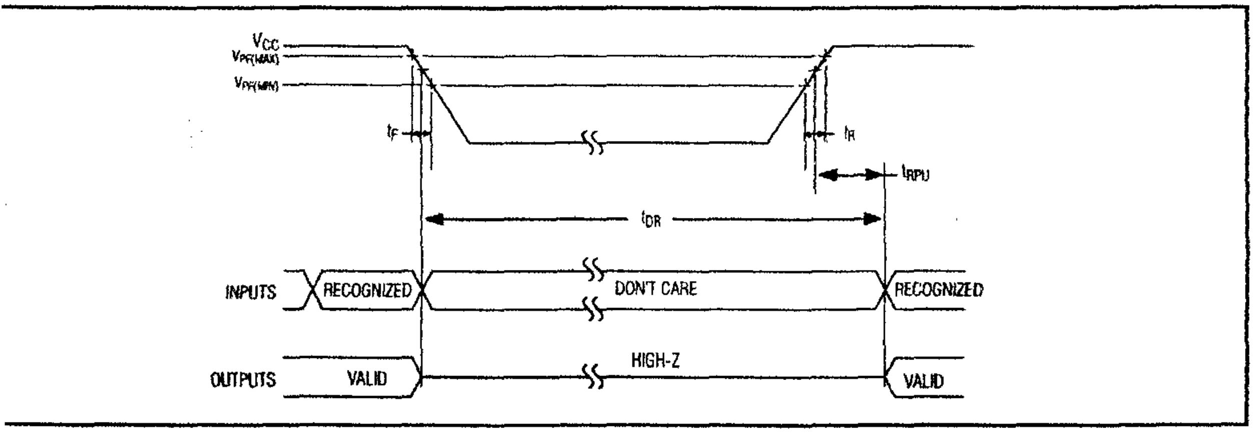
Intel Bus Read Timing



IRQ Release Delay Timing



Power-Up/Power-Down Timing



Real-Time Clock

POWER-UP/POWER-DOWN CHARACTERISTICS

(T_A = -40°C to +85°C) (Note 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Recovery at Power-Up	t_{RPU}		20		200	ms
V_{CC} Fall Time; $V_{PF(MAX)}$ to $V_{PF(MIN)}$	t_F		300			µs
V_{CC} Rise Time; $V_{PF(MIN)}$ to $V_{PF(MAX)}$	t_R		0			µs

DATA RETENTION

(T_A = +25°C)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Expected Data Retention	t_{DR}		10			years

CAPACITANCE

(T_A = +25°C) (Note 9)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Capacitance on All Input Pins Except X1 and X2	C_{IN}				5	pF
Capacitance on $\overline{IRQ}$, SQW, and DQ Pins	C_{IO}				7	pF

AC TEST CONDITIONS

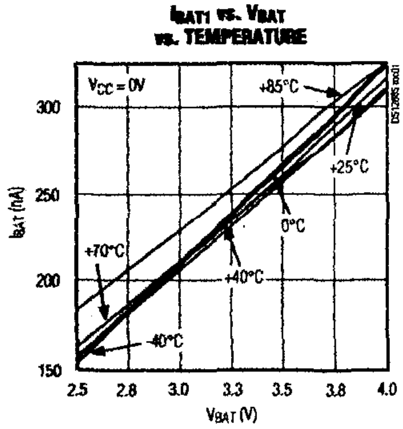
PARAMETER	TEST CONDITIONS
Input Pulse Levels	0 to 3.0V
Output Load Including Scope and Jig	50pF + 1TTL Gate
Input and Output Timing Measurement Reference Levels	Input/Output: V_{IL} maximum and V_{IH} minimum
Input-Pulse Rise and Fall Times	5ns

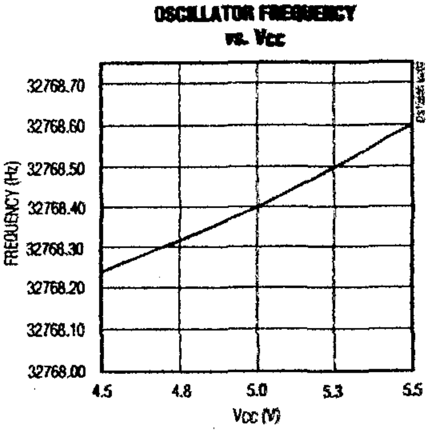
WARNING: Negative undershoots below -0.3V while the part is in battery-backed mode may cause loss of data.

Note 1: RTC modules can be successfully processed through conventional wave-soldering techniques as long as temperature exposure to the lithium energy source contained within does not exceed +85°C. However, post-solder cleaning with water-washing techniques is acceptable, provided that ultrasonic vibrations are not used to prevent crystal damage.
Note 2: Limits at -40°C are guaranteed by design and not production tested.
Note 3: All voltages are referenced to ground.
Note 4: All outputs are open.
Note 5: Specified with $\overline{CS}$ = DS = R/$\overline{W}$ = $\overline{RESET}$ = V_{CC}; MOT, AS, AD0–AD7 = 0; V_{BACKUP} open.
Note 6: Applies to the AD0 to AD7 pins, the $\overline{IRQ}$ pin, and the SQW pin when each is in a high-impedance state.
Note 7: The MOT pin has an internal 20kΩ pulldown.
Note 8: Measured with a 32.768kHz crystal attached to X1 and X2.
Note 9: Guaranteed by design. Not production tested.
Note 10: Measured with a 50pF capacitance load.

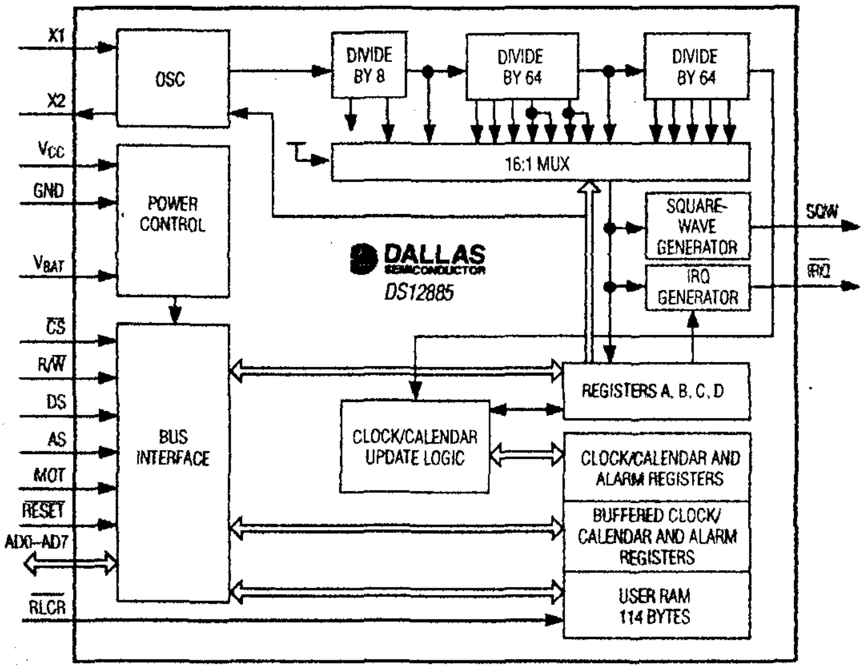
Typical Operating Characteristics

(V_{CC} = +5.0V, T_A = +25°C, unless otherwise noted.)





Functional Diagram



Pin Description

PIN				NAME	FUNCTION
SO, PDIP	EDIP	PLCC	TQFP		
1	1	2	29	MOT	Motorola or Intel Bus Timing Selector. This pin selects one of two bus types. When connected to V_{CC}, Motorola bus timing is selected. When connected to GND or left disconnected, Intel bus timing is selected. The pin has an internal pulldown resistor.
2	—	3	30	X1	Connections for Standard 32.768kHz Quartz Crystal. The internal oscillator circuitry is designed for operation with a crystal having a 6pF specified load capacitance (C_L). Pin X1 is the input to the oscillator and can optionally be connected to an external 32.768kHz oscillator. The output of the internal oscillator, pin X2, is floated if an external oscillator is connected to pin X1.
3	—	4	31	X2	
4–11	4–11	5–10, 12, 14	1, 2, 3, 5, 7, 8, 9, 11	AD0–AD7	Multiplexed, Bidirectional Address/Data Bus. The addresses are presented during the first portion of the bus cycle and latched into the device by the falling edge of AS. Write data is latched by the falling edge of DS (Motorola timing) or the rising edge of R/$\overline{W}$ (Intel timing). In a read cycle, the device outputs data during the latter portion of DS (DS and R/$\overline{W}$ high for Motorola timing, DS low and R/$\overline{W}$ high for Intel timing). The read cycle is terminated and the bus returns to a high-impedance state as DS transitions low in the case of Motorola timing or as DS transitions high in the case of Intel timing.
12, 16	12	15, 20	12, 17	GND	Ground
13	13	16	13	$\overline{CS}$	Active-Low Chip-Select Input. The chip-select signal must be asserted low for a bus cycle in the device to be accessed. $\overline{CS}$ must be kept in the active state during DS and AS for Motorola timing and during DS and R/$\overline{W}$ for Intel timing. Bus cycles that take place without asserting $\overline{CS}$ will latch addresses, but no access occurs. When V_{CC} is below V_{PF} volts, the device inhibits access by internally disabling the $\overline{CS}$ input. This action protects the RTC data and the RAM data during power outages.
14	14	17	14	AS	Address Strobe Input. A positive-going address-strobe pulse serves to *demultiplex the bus. The falling edge of AS causes the address to be latched* within the device. The next rising edge that occurs on the AS bus clears the address regardless of whether $\overline{CS}$ is asserted. An address strobe must immediately precede each write or read access. If a write or read is performed with $\overline{CS}$ deasserted, another address strobe must be performed prior to a read or write access with $\overline{CS}$ asserted.
15	15	19	16	R/$\overline{W}$	Read/Write Input. The R/$\overline{W}$ pin has two modes of operation. When the MOT pin is connected to V_{CC} for Motorola timing, R/$\overline{W}$ is at a level that indicates whether the current cycle is a read or write. A read cycle is indicated with a high level on R/$\overline{W}$ while DS is high. A write cycle is indicated when R/$\overline{W}$ is low during DS. When the MOT pin is connected to GND for Intel timing, the R/$\overline{W}$ signal is an active-low signal. In this mode, the R/$\overline{W}$ pin operates in a similar fashion as the write-enable signal ($\overline{WE}$) on generic RAMs. Data are latched on the rising edge of the signal.

Pin Description (continued)

PIN				NAME	FUNCTION
SO, PDIP	EDIP	PLCC	TQFP		
22	2, 3, 16, 20, 21, 22	1, 11, 13, 18, 26	4, 6, 10, 15, 20, 23, 25, 27, 32	N.C.	No Connection. This pin should remain unconnected. Pin 21 is $\overline{RCLR}$ for the *DS12887A/DS12C887A. On the EDIP, these pins are missing by design.*
17	17	21	18	DS	Data Strobe or Read Input. The DS pin has two modes of operation depending on the level of the MOT pin. When the MOT pin is connected to V_{CC}, Motorola bus timing is selected. In this mode, DS is a positive pulse during the latter portion of the bus cycle and is called data strobe. During read cycles, DS signifies the time that the device is to drive the bidirectional bus. In write cycles, the trailing edge of DS causes the device to latch the written data. When the MOT pin is connected to GND, Intel bus timing is selected. DS identifies the time period when the device drives the bus with read data. In this mode, the DS pin operates in a similar fashion as the output-enable ($\overline{OE}$) signal on a generic RAM.
18	18	22	19	$\overline{RESET}$	Active-Low Reset Input. The $\overline{RESET}$ pin has no effect on the clock, calendar, or RAM. On power-up, the $\overline{RESET}$ pin can be held low for a time to allow the power supply to stabilize. The amount of time that $\overline{RESET}$ is held low is dependent on the application. However, if $\overline{RESET}$ is used on power-up, the time $\overline{RESET}$ is low should exceed 200ms to ensure that the internal timer that controls the device on power-up has timed out. When $\overline{RESET}$ is low and V_{CC} is above V_{PF}, the following occurs: *A. Periodic interrupt-enable (PIE) bit is cleared to 0.* B. Alarm interrupt-enable (AIE) bit is cleared to 0. C. Update-ended interrupt-enable (UIE) bit is cleared to 0. D. Periodic-interrupt flag (PF) bit is cleared to 0. E. Alarm-interrupt flag (AF) bit is cleared to 0. F. Update-ended interrupt flag (UF) bit is cleared to 0. G. Interrupt-request status flag (IRQF) bit is cleared to 0. H. $\overline{IRQ}$ pin is in the high-impedance state. I. The device is not accessible until $\overline{RESET}$ is returned high. J. Square-wave output-enable (SQWE) bit is cleared to 0. In a typical application, $\overline{RESET}$ can be connected to V_{CC}. This connection allows the device to go in and out of power fail without affecting any of the control registers.

Pin Description (continued)

PIN				NAME	FUNCTION
SO, PDIP	EDIP	PLCC	TQFP		
19	19	23	21	$\overline{\text{IRQ}}$	Active-Low Interrupt Request Output. The $\overline{\text{IRQ}}$ pin is an active-low output of the device that can be used as an interrupt input to a processor. The $\overline{\text{IRQ}}$ output remains low as long as the status bit causing the interrupt is present and the corresponding interrupt-enable bit is set. The processor program normally reads the C register to clear the $\overline{\text{IRQ}}$ pin. The $\overline{\text{RESET}}$ pin also clears pending interrupts. When no interrupt conditions are present, the $\overline{\text{IRQ}}$ level is in the high-impedance state. Multiple interrupting devices can be connected to an $\overline{\text{IRQ}}$ bus, provided that they are all open drain. The $\overline{\text{IRQ}}$ pin is an open-drain output and requires an external pullup resistor to V_{CC}.
20	—	24	22	V_{BAT}	Connection for a Primary Battery. (DS12885 Only.) Battery voltage must be held between the minimum and maximum limits for proper operation. If a backup supply is not supplied, V_{BAT} must be grounded. Connect the battery directly to the V_{BAT} pin. Diodes in series between the V_{BAT} pin and the battery may prevent proper operation. UL recognized to ensure against reverse charging when used with a lithium battery.
21	21 (DS12887A/ DS12C887A)	25	24	$\overline{\text{RCLR}}$	Active-Low RAM Clear. The $\overline{\text{RCLR}}$ pin is used to clear (set to logic 1) all the general-purpose RAM, but does not affect the RAM associated with the RTC. To clear the RAM, $\overline{\text{RCLR}}$ must be forced to an input logic 0 during battery-backup mode when V_{CC} is not applied. The $\overline{\text{RCLR}}$ function is designed to be used through a human interface (shorting to ground manually or by a switch) and not to be driven with external buffers. This pin is internally pulled up. Do not use an external pullup resistor on this pin.
23	23	27	26	SQW	Square-Wave Output. The SQW pin can output a signal from one of 13 taps provided by the 15 internal divider stages of the RTC. The frequency of the SQW pin can be changed by programming Register A, as shown in Table 1. The SQW signal can be turned on and off using the SQWE bit in Register B. The SQW signal is not available when V_{CC} is less than V_{PF}.
24	24	28	28	V_{CC}	DC Power Pin for Primary Power Supply. When V_{CC} is applied within normal limits, the device is fully accessible and data can be written and read. When V_{CC} is below V_{PF} reads and writes are inhibited.

Detailed Description

'he DS12885 family of RTCs provide 14 bytes of real-me clock/calendar, alarm, and control/status registers ınd 114 bytes (113 bytes for DS12C887 and)S12C887A) of nonvolatile, battery-backed static RAM. \ time-of-day alarm, three maskable interrupts with a :ommon interrupt output, and a programmable square-vave output are available. The devices also operate in ither 24-hour or 12-hour format with an AM/PM indica-or. A precision temperature-compensated circuit moni-ors the status of V_{CC}. If a primary power-supply failure s detected, the devices automatically switch to a back-ıp supply. The backup supply input supports a primary ›attery, such as lithium coin cell. The devices are ıccessed through a multiplexed address/data bus that :upports Intel and Motorola modes.

Oscillator Circuit

'he DS12885 uses an external 32.768kHz crystal. The ›scillator circuit does not require any external resistors ır capacitors to operate. Table 1 specifies several crys-al parameters for the external crystal. Figure 1 shows a ınctional schematic of the oscillator circuit. An enable ›it in the control register controls the oscillator.)scillator startup times are highly dependent upon :rystal characteristics, PC board leakage, and layout. ligh ESR and excessive capacitive loads are the major :ontributors to long startup times. A circuit using a :rystal with the recommended characteristics and ›roper layout usually starts within one second.

ın external 32.768kHz oscillator can also drive the)S12885. In this configuration, the X1 pin is connected ı the external oscillator signal and the X2 pin is floated.

Table 1. Crystal Specifications*

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS
Nominal Frequency	f_O		32.768		kHz
Series Resistance	ESR			50	kΩ
Load Capacitance	C_L		6		pF

**The crystal, traces, and crystal input pins should be isolated from RF generating signals. Refer to* Application Note 58: Crystal Considerations for Dallas Real-Time Clocks *for additional specifications.*

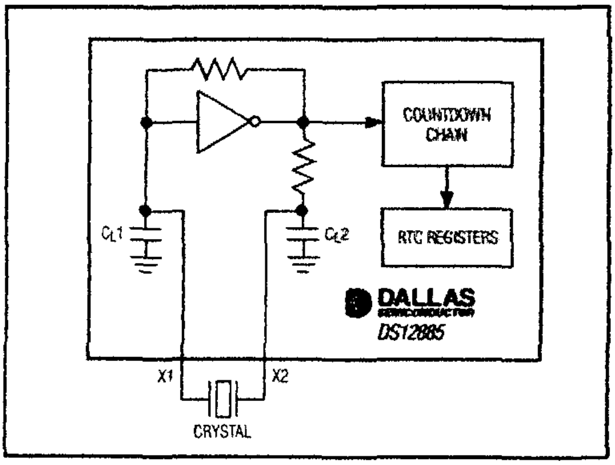


Figure 1. Oscillator Circuit Showing Internal Bias Network

Real-Time Clock

Clock Accuracy

The accuracy of the clock is dependent upon the accuracy of the crystal and the accuracy of the match between the capacitive load of the oscillator circuit and the capacitive load for which the crystal was trimmed. Additional error is added by crystal frequency drift caused by temperature shifts. External circuit noise coupled into the oscillator circuit can result in the clock running fast. Figure 2 shows a typical PC board layout for isolation of the crystal and oscillator from noise. Refer to *Application Note 58: Crystal Considerations with Dallas Real-Time Clocks* for more detailed information.

Clock Accuracy for DS12887, DS12887A, DS12C887, DS12C887A Only

The encapsulated DIP modules are trimmed at the factory to an accuracy of ±1 minute per month at +25°C.

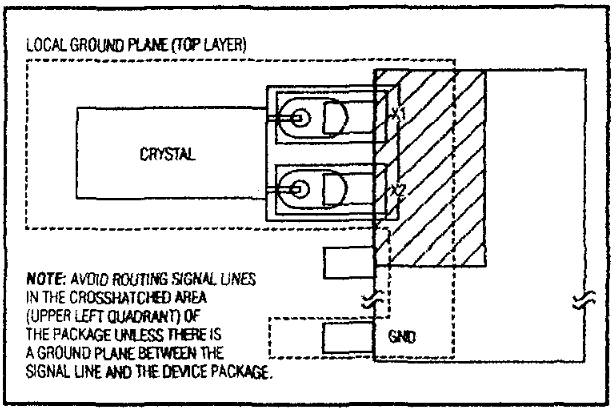
Power-Down/Power-Up Considerations

The real-time clock continues to operate, and the RAM, time, calendar, and alarm memory locations remain nonvolatile regardless of the V_{CC} input level. V_{BAT} must remain within the minimum and maximum limits when V_{CC} is not applied. When V_{CC} is applied and exceeds V_{PF} (power-fail trip point), the device becomes accessible after t_{REC}—if the oscillator is running and the oscillator countdown chain is not in reset (Register A). This time allows the system to stablize after power is applied. If the oscillator is not enabled, the oscillator-enable bit is enabled on power-up, and the device becomes immediately accessible.

Time, Calendar, and Alarm Locations

The time and calendar information is obtained by reading the appropriate register bytes. The time, calendar, and alarm are set or initialized by writing the appropriate register bytes. Invalid time or date entries result in undefined operation. The contents of the 10 time, calendar, and alarm bytes can be either binary or binary-coded decimal (BCD) format.

The day-of-week register increments at midnight, incrementing from 1 through 7. The day-of-week register is used by the daylight saving function, so the value 1 is defined as Sunday. The date at the end of the month is automatically adjusted for months with fewer than 31 days, including correction for leap years.



NOTE: AVOID ROUTING SIGNAL LINES IN THE CROSSHATCHED AREA (UPPER LEFT QUADRANT) OF THE PACKAGE UNLESS THERE IS A GROUND PLANE BETWEEN THE SIGNAL LINE AND THE DEVICE PACKAGE.

Figure 2. Layout Example

Before writing the internal time, calendar, and alarm registers, the SET bit in Register B should be written to logic 1 to prevent updates from occurring while access is being attempted. In addition to writing the 10 time, calendar, and alarm registers in a selected format (binary or BCD), the data mode bit (DM) of Register B must be set to the appropriate logic level. All 10 time, calendar, and alarm bytes must use the same data mode. The SET bit in Register B should be cleared after the data mode bit has been written to allow the RTC to update the time and calendar bytes. Once initialized, the RTC makes all updates in the selected mode. The data mode cannot be changed without reinitializing the 10 data bytes. Tables 2A and 2B show the BCD and binary formats of the time, calendar, and alarm locations.

The 24-12 bit cannot be changed without reinitializing the hour locations. When the 12-hour format is selected, the higher-order bit of the hours byte represents PM when it is logic 1. The time, calendar, and alarm bytes are always accessible because they are double-buffered. Once per second the seven bytes are advanced by one second and checked for an alarm condition.

If a read of the time and calendar data occurs during an update, a problem exists where seconds, minutes, hours, etc., may not correlate. The probability of reading incorrect time and calendar data is low. Several methods of avoiding any possible incorrect time and calendar reads are covered later in this text.

he three alarm bytes can be used in two ways. First, vhen the alarm time is written in the appropriate hours, ninutes, and seconds alarm locations, the alarm interupt is initiated at the specified time each day, if the larm-enable bit is high. In this mode, the "0" bits in the larm registers and the corresponding time registers *nust always be written to 0* (Table 2A and 2B). Writing ne 0 bits in the alarm and/or time registers to 1 can esult in undefined operation.

he second use condition is to insert a "don't care" tate in one or more of the three alarm bytes. The don't-are code is any hexadecimal value from C0 to FF. The wo most significant bits of each byte set the don't-care condition when at logic 1. An alarm is generated each hour when the don't-care bits are set in the hours byte. Similarly, an alarm is generated every minute with don't-care codes in the hours and minute alarm bytes. The don't-care codes in all three alarm bytes create an interrupt every second.

All 128 bytes can be directly written or read, except for the following:

1) Registers C and D are read-only.
2) Bit 7 of register A is read-only.
3) The MSB of the seconds byte is read-only.

able 2A. Time, Calendar, and Alarm Data Modes—BCD Mode (DM = 0)

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00H	0	10 Seconds			Seconds				Seconds	00–59
01H	0	10 Seconds			Seconds				Seconds Alarm	00–59
02H	0	10 Minutes			Minutes				Minutes	00–59
03H	0	10 Minutes			Minutes				Minutes Alarm	00–59
04H	AM/PM	0	0	10 Hours	Hours				Hours	1–12 +AM/PM
	0		10 Hours							00–23
05H	AM/PM	0	0	10 Hours	Hours				Hours Alarm	1–12 +AM/PM
	0		10 Hours							00–23
06H	0	0	0	0	0	Day			Day	01–07
07H	0	0	10 Date		Date				Date	01–31
08H	0	0	0	10 Months	Month				Month	01–12
09H	10 Years				Year				Year	00–99
0AH	UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0	Control	—
0BH	SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE	Control	—
0CH	IRQF	PF	AF	UF	0	0	0	0	Control	—
0DH	VRT	0	0	0	0	0	0	0	Control	—
0EH-31H	X	X	X	X	X	X	X	X	RAM	—
32H	10 Century				Century				Century*	00–99
33H-7FH	X	X	X	X	X	X	X	X	RAM	—

' = Read/Write Bit.

DS12C887, DS12C887A only. General-purpose RAM on DS12885, DS12887, and DS12887A.

***lote:** Unless otherwise specified, the state of the registers is not defined when power is first applied. Except for the seconds regis-r, 0 bits in the time and date registers can be written to 1, but may be modified when the clock updates. 0 bits should always be ritten to 0 except for alarm mask bits.*

Table 2B. Time, Calendar, and Alarm Data Modes—Binary Mode (DM = 1)

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00H	0	0	Seconds						Seconds	00–3B
01H	0	0	Seconds						Seconds Alarm	00–3B
02H	0	0	Minutes						Minutes	00–3B
03H	0	0	Minutes						Minutes Alarm	00–3B
04H	AM/PM	0	0	0	Hours				Hours	01–0C +AM/PM 00–17
	0			Hours						
05H	AM/PM	0	0	0	Hours				Hours Alarm	01–0C +AM/PM 00–17
	0			Hours						
06H	0	0	0	0	0		Day		Day	01–07
07H	0	0	0	Date					Date	01–1F
08H	0	0	0	0	Month				Month	01–0C
09H	0	Year							Year	00–63
0AH	UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0	Control	—
0BH	SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE	Control	—
0CH	IRQF	PF	AF	UF	0	0	0	0	Control	—
0DH	VRT	0	0	0	0	0	0	0	Control	—
0EH-31H	X	X	X	X	X	X	X	X	RAM	—
32H	N/A				N/A				Century*	—
33H-7FH	X	X	X	X	X	X	X	X	RAM	—

X = Read/Write Bit.

*****DS12C887, DS12C887A only. General-purpose RAM on DS12885, DS12887, and DS12887A.*

***Note:** Unless otherwise specified, the state of the registers is not defined when power is first applied. Except for the seconds register, 0 bits in the time and date registers can be written to 1, but may be modified when the clock updates. 0 bits should always be written to 0 except for alarm mask bits.*

Control Registers

'he real-time clocks have four control registers that are ıccessible at all times, even during the update cycle.

Control Register A

ISB							LSB
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
UIP	DV2	DV1	DV0	RS3	RS2	RS1	RS0

ılt 7: Update-In-Progress (UIP). This bit is a status ag that can be monitored. When the UIP bit is a 1, the pdate transfer occurs soon. When UIP is a 0, the pdate transfer does not occur for at least 244µs. The me, calendar, and alarm information in RAM is fully vailable for access when the UIP bit is 0. The UIP bit is ead-only and is not affected by $\overline{\text{RESET}}$. Writing the ET bit in Register B to a 1 inhibits any update transfer nd clears the UIP status bit.

ılts 6, 5, and 4: DV2, DV1, DV0. These three bits are sed to turn the oscillator on or off and to reset the ountdown chain. A pattern of 010 is the only combinaon of bits that turn the oscillator on and allow the RTC keep time. A pattern of 11x enables the oscillator but *olds the countdown chain in reset. The next update* ccurs at 500ms after a pattern of 010 is written to DV0, V1, and DV2.

Bits 3 to 0: Rate Selector (RS3, RS2, RS1, RS0). These four rate-selection bits select one of the 13 taps on the 15-stage divider or disable the divider output. The tap selected can be used to generate an output square wave (SQW pin) and/or a periodic interrupt. The user can do one of the following:

1) Enable the interrupt with the PIE bit;
2) Enable the SQW output pin with the SQWE bit;
3) Enable both at the same time and the same rate; or
4) Enable neither.

Table 3 lists the periodic interrupt rates and the square-wave frequencies that can be chosen with the RS bits. These four read/write bits are not affected by $\overline{\text{RESET}}$.

Real-Time Clock

Control Register B

MSB **LSB**

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
SET	PIE	AIE	UIE	SQWE	DM	24/12	DSE

Bit 7: SET. When the SET bit is 0, the update transfer functions normally by advancing the counts once per second. When the SET bit is written to 1, any update transfer is inhibited, and the program can initialize the time and calendar bytes without an update occurring in the midst of initializing. Read cycles can be executed in a similar manner. SET is a read/write bit and is not affected by $\overline{\text{RESET}}$ or internal functions of the device.

Bit 6: Periodic Interrupt Enable (PIE). The PIE bit is a read/write bit that allows the periodic interrupt flag (PF) bit in Register C to drive the $\overline{\text{IRQ}}$ pin low. When the PIE bit is set to 1, periodic interrupts are generated by driving the $\overline{\text{IRQ}}$ pin low at a rate specified by the RS3–RS0 bits of Register A. A 0 in the PIE bit blocks the $\overline{\text{IRQ}}$ output from being driven by a periodic interrupt, but the PF bit is still set at the periodic rate. PIE is not modified by any internal device functions, but is cleared to 0 on $\overline{\text{RESET}}$.

Bit 5: Alarm Interrupt Enable (AIE). This bit is a read/write bit that, when set to 1, permits the alarm flag (AF) bit in Register C to assert $\overline{\text{IRQ}}$. An alarm interrupt occurs for each second that the three time bytes equal the three alarm bytes, including a don't-care alarm code of binary 11XXXXXX. The AF bit does not initiate the $\overline{\text{IRQ}}$ signal when the AIE bit is set to 0. The internal functions of the device do not affect the AIE bit, but is cleared to 0 on $\overline{\text{RESET}}$.

Bit 4: Update-Ended Interrupt Enable (UIE). This bit is a read/write bit that enables the update-end flag (UF) bit in Register C to assert $\overline{\text{IRQ}}$. The $\overline{\text{RESET}}$ pin going low or the SET bit going high clears the UIE bit.

The internal functions of the device do not affect the UIE bit, but is cleared to 0 on $\overline{\text{RESET}}$.

Bit 3: Square-Wave Enable (SQWE). When this bit is set to 1, a square-wave signal at the frequency set by the rate-selection bits RS3–RS0 is driven out on the SQW pin. When the SQWE bit is set to 0, the SQW pin is held low. SQWE is a read/write bit and is cleared by $\overline{\text{RESET}}$. SQWE is low if disabled, and is high impedance when V_{CC} is below V_{PF}. SQWE is cleared to 0 on $\overline{\text{RESET}}$.

Bit 2: Data Mode (DM). This bit indicates whether time and calendar information is in binary or BCD format. The DM bit is set by the program to the appropriate format and can be read as required. This bit is not modified by internal functions or $\overline{\text{RESET}}$. A 1 in DM signifies binary data, while a 0 in DM specifies BCD data.

Bit 1: 24/12. The 24/12 control bit establishes the format of the hours byte. A 1 indicates the 24-hour mode and a 0 indicates the 12-hour mode. This bit is read/write and is not affected by internal functions or $\overline{\text{RESET}}$.

Bit 0: Daylight Saving Enable (DSE). This bit is a read/write bit that enables two daylight saving adjustments when DSE is set to 1. On the first Sunday in April, the time increments from 1:59:59 AM to 3:00:00 AM. On the last Sunday in October when the time first reaches 1:59:59 AM, it changes to 1:00:00 AM. When DSE is enabled, the internal logic test for the first/last Sunday condition at midnight. If the DSE bit is not set when the test occurs, the daylight saving function does not operate correctly. These adjustments do not occur when the DSE bit is 0. This bit is not affected by internal functions or $\overline{\text{RESET}}$.

Control Register C

ISB LSB

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
IRQF	PF	AF	UF	0	0	0	0

it 7: Interrupt Request Flag (IRQF). This bit is set to when any of the following are true:

PF = PIE = 1

AF = AIE = 1

UF = UIE = 1

ny time the IRQF bit is 1, the $\overline{\text{IRQ}}$ pin is driven low. nis bit can be cleared by reading Register C or with a $\overline{\text{ESET}}$.

it 6: Periodic Interrupt Flag (PF). This bit is readnly and is set to 1 when an edge is detected on the *elected tap of the divider chain. The RS3 through RS0* its establish the periodic rate. PF is set to 1 independent of the state of the PIE bit. When both PF and PIE re 1s, the $\overline{\text{IRQ}}$ signal is active and sets the IRQF bit. his bit can be cleared by reading Register C or with a $\overline{\text{ESET}}$.

Bit 5: Alarm Interrupt Flag (AF). A 1 in the AF bit indicates that the current time has matched the alarm time. If the AIE bit is also 1, the $\overline{\text{IRQ}}$ pin goes low and a 1 appears in the IRQF bit. This bit can be cleared by reading Register C or with a $\overline{\text{RESET}}$.

Bit 5: Update-Ended Interrupt Flag (UF). This bit is *set after each update cycle. When the UIE bit is set to* 1, the 1 in UF causes the IRQF bit to be a 1, which asserts the $\overline{\text{IRQ}}$ pin. This bit can be cleared by reading Register C or with a $\overline{\text{RESET}}$.

Bits 3 to 0: Unused. These bits are unused in Register *C. These bits always read 0 and cannot be written.*

Control Register D

ISB LSB

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
VRT	0	0	0	0	0	0	0

it 7: Valid RAM and Time (VRT). This bit indicates ıe condition of the battery connected to the V_{BAT} pin. ıis bit is not writeable and should always be 1 when ıad. If a 0 is ever present, an exhausted internal lithi*n energy source is indicated and both the contents of* the RTC data and RAM data are questionable. This bit is unaffected by $\overline{\text{RESET}}$.

Bits 6 to 0: Unused. The remaining bits of Register D are not usable. They cannot be written and they always read 0.

Century Register (DS12C887/DS12C887A Only)

The century register at location 32h is a BCD register designed to automatically load the BCD value 20 as the year register changes from 99 to 00. The MSB of this register is not affected when the load of 20 occurs, and remains at the value written by the user.

Nonvolatile RAM (NV RAM)

The general-purpose NV RAM bytes are not dedicated to any special function within the device. They can be used by the processor program as battery-backed memory and are fully available during the update cycle.

Interrupts

The RTC family includes three separate, fully automatic sources of interrupt for a processor. The alarm interrupt can be programmed to occur at rates from once per second to once per day. The periodic interrupt can be selected for rates from 500ms to 122µs. The update-ended interrupt can be used to indicate to the program that an update cycle is complete. Each of these independent interrupt conditions is described in greater detail in other sections of this text.

The processor program can select which interrupts, if any, are to be used. Three bits in Register B enable the interrupts. Writing a logic 1 to an interrupt-enable bit permits that interrupt to be initiated when the event occurs. A 0 in an interrupt-enable bit prohibits the $\overline{\text{IRQ}}$ pin from being asserted from that interrupt condition. If an interrupt flag is already set when an interrupt is enabled, $\overline{\text{IRQ}}$ is immediately set at an active level, although the interrupt initiating the event may have occurred earlier. As a result, there are cases where the program should clear such earlier initiated interrupts before first enabling new interrupts.

When an interrupt event occurs, the relating flag bit is set to logic 1 in Register C. These flag bits are set independent of the state of the corresponding enable bit in Register B. The flag bit can be used in a polling mode without enabling the corresponding enable bits. The interrupt flag bit is a status bit that software can interrogate as necessary. When a flag is set, an indication is given to software that an interrupt event has occurred since the flag bit was last read; however, care should be taken when using the flag bits as they are cleared each time Register C is read. Double latching is included with Register C so that bits that are set remain stable throughout the read cycle. All bits that are set (high) are cleared when read, and new interrupts that are pending during the read cycle are held until after the cycle is completed. One, two, or three bits can be set when reading Register C. Each used flag bit should be examined when Register C is read to ensure that no interrupts are lost.

The second flag bit method is used with fully enabled interrupts. When an interrupt flag bit is set and the corresponding interrupt-enable bit is also set, the $\overline{\text{IRQ}}$ pin is asserted low. $\overline{\text{IRQ}}$ is asserted as long as at least one of the three interrupt sources has its flag and enable bits set. The IRQF bit in Register C is a 1 whenever the $\overline{\text{IRQ}}$ pin is driven low. Determination that the RTC initiated an interrupt is accomplished by reading Register C. A logic 1 in bit 7 (IRQF bit) indicates that one or more interrupts have been initiated by the device. The act of reading Register C clears all active flag bits and the IRQF bit.

Oscillator Control Bits

When the DS12887, DS12887A, DS12C887, and DS12C887A are shipped from the factory, the internal oscillator is turned off. This prevents the lithium energy cell from being used until the device is installed in a system.

A pattern of 010 in bits 4 to 6 of Register A turns the oscillator on and enables the countdown chain. A pattern of 11x (DV2 = 1, DV1 = 1, DV0 = X) turns the oscillator on, but holds the countdown chain of the oscillator in reset. All other combinations of bits 4 to 6 keep the oscillator off.

Square-Wave Output Selection

Thirteen of the 15 divider taps are made available to a 1-of-16 multiplexer, as shown in the functional diagram. The square-wave and periodic-interrupt generators share the output of the multiplexer. The RS0–RS3 bits in Register A establish the output frequency of the multiplexer (see Table 1). Once the frequency is selected, the output of the SQW pin can be turned on and off under program control with the square-wave enable bit, SQWE.

Periodic Interrupt Selection

The periodic interrupt causes the $\overline{\text{IRQ}}$ pin to go to an active state from once every 500ms to once every 122µs. This function is separate from the alarm interrupt, which can be output from once per second to once per day. The periodic interrupt rate is selected using the same Register A bits that select the square-wave frequency (Table 1). Changing the Register A bits affects the square-wave frequency and the periodic-interrupt output. However, each function has a separate enable bit in Register B. The SQWE bit controls the square-wave output. Similarly, the PIE bit in Register B enables the periodic interrupt. The periodic interrupt can be used with software counters to measure inputs, create output intervals, or await the next needed software function.

'able 3. Periodic Interrupt Rate and ;quare-Wave Output Frequency

SELECT BITS REGISTER A				t_{PI} PERIODIC INTERRUPT RATE	SQW OUTPUT FREQUENCY
RS3	RS2	RS1	RS0		
0	0	0	0	None	None
0	0	0	1	3.90625ms	256Hz
0	0	1	0	7.8125ms	128Hz
0	0	1	1	122.070µs	8.192kHz
0	1	0	0	244.141µs	4.096kHz
0	1	0	1	488.281µs	2.048kHz
0	1	1	0	976.5625µs	1.024kHz
0	1	1	1	1.953125ms	512Hz
1	0	0	0	3.90625ms	256Hz
1	0	0	1	7.8125ms	128Hz
1	0	1	0	15.625ms	64Hz
1	0	1	1	31.25ms	32Hz
1	1	0	0	62.5ms	16Hz
1	1	0	1	125ms	8Hz
1	1	1	0	250ms	4Hz
1	1	1	1	500ms	2Hz

Update Cycle

he device executes an update cycle once per second ›gardless of the SET bit in Register B. When the SET it in Register B is set to 1, the user copy of the double-uffered time, calendar, and alarm bytes is frozen and oes not update as the time increments. However, the me countdown chain continues to update the internal opy of the buffer. This feature allows time to maintain ccuracy independent of reading or writing the time, alendar, and alarm buffers, and also guarantees that me and calendar information is consistent. The update ycle also compares each alarm byte with the corresponding time byte and issues an alarm if a match or if a don't-care code is present in all three positions.

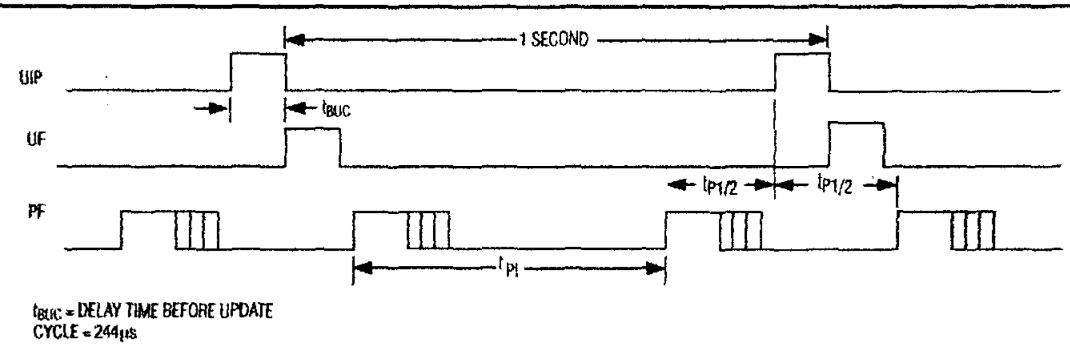
There are three methods that can handle RTC access that avoid any possibility of accessing inconsistent time and calendar data. The first method uses the update-ended interrupt. If enabled, an interrupt occurs after every update cycle that indicates over 999ms is available to read valid time and date information. If this interrupt is used, the IRQF bit in Register C should be cleared before leaving the interrupt routine.

A second method uses the update-in-progress bit (UIP) in Register A to determine if the update cycle is in progress. The UIP bit pulses once per second. After the UIP bit goes high, the update transfer occurs 244µs later. If a low is read on the UIP bit, the user has at least 244µs before the time/calendar data is changed. Therefore, the user should avoid interrupt service routines that would cause the time needed to read valid time/calendar data to exceed 244µs.

The third method uses a periodic interrupt to determine if an update cycle is in progress. The UIP bit in Register A is set high between the setting of the PF bit in Register C (Figure 3). Periodic interrupts that occur at a rate greater than t_{BUC} allow valid time and date information to be reached at each occurrence of the periodic interrupt. The reads should be complete within one ($t_{PI/2}$ + t_{BUC}) to ensure that data is not read during the update cycle.

Handling, PC Board Layout, and Assembly

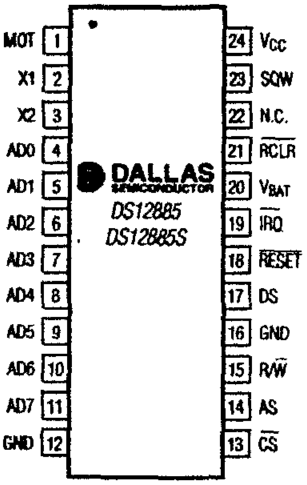
The EDIP module can be successfully processed through conventional wave-soldering techniques so long as temperature exposure to the lithium energy source does not exceed +85°C. Post-solder cleaning with water-washing techniques is acceptable, provided that ultrasonic vibration is not used. Such cleaning can damage the crystal.



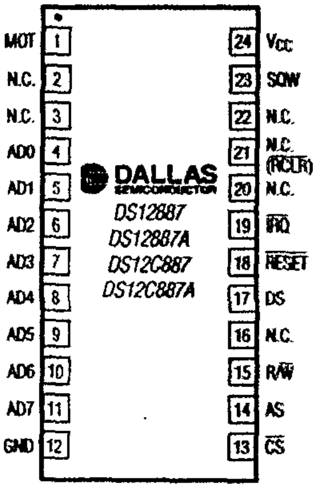
gure 3. UIP and Periodic Interrupt Timing

Pin Configurations

TOP VIEW

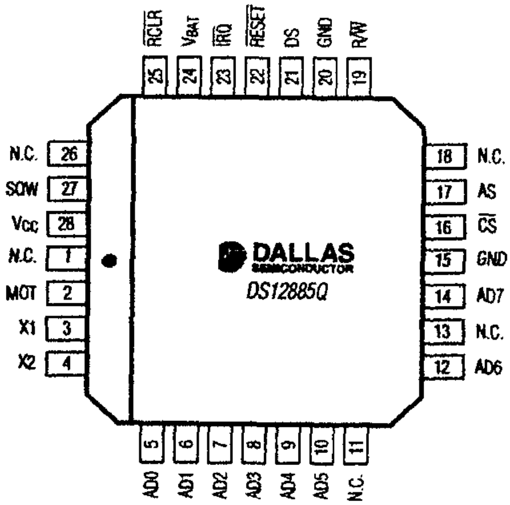


SO, PDIP



EDIP

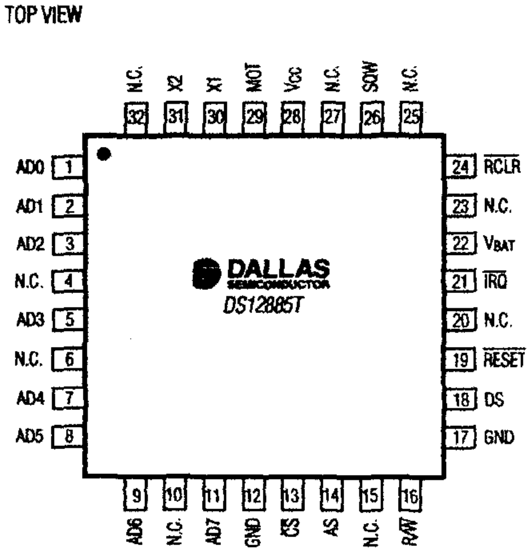
( ) FOR THE DS12887A/DS12C887A.



PLCC

NOTE: THE DS12887A AND DS12C887A CANNOT BE STORED OR SHIPPED IN CONDUCTIVE MATERIAL THAT WILL GIVE A CONTINUITY PATH BETWEEN THE RAM CLEAR PIN AND GROUND.

Pin Configurations (continued)



TQFP

Thermal Information

PACKAGE	THETA-JA (°C/W)	THETA-JC (°C/W)
PDIP	75	30
SO	105	22
PLCC	95	25

Ordering Information

PART	TEMP RANGE	PIN-PACKAGE	TOP MARK*
DS12885	0°C to +70°C	24 PDIP	DS12885
DS12885N	-40°C to +85°C	24 PDIP	DS12885N
DS12885Q	0°C to +70°C	28 PLCC	DS12885Q
DS12885Q+	0°C to +70°C	28 PLCC	DS12885Q
DS12885QN	-40°C to +85°C	28 PLCC	DS12885Q
DS12885QN+	-40°C to +85°C	28 PLCC	DS12885Q
DS12885S	0°C to +70°C	24 SO (300 mils)	DS12885S
DS12885S+	0°C to +70°C	24 SO (300 mils)	DS12885S
DS12885SN	-40°C to +85°C	24 SO (300 mils)	DS12885S
DS12885SN+	-40°C to +85°C	24 SO (300 mils)	DS12885S
DS12885T	0°C to +70°C	32 TQFP	DS12885T
DS12885TN	-40°C to +85°C	32 TQFP	DS12885T
DS12887	0°C to +70°C	24 EDIP	DS12887
DS12887A	0°C to +70°C	24 EDIP	DS12887A
DS12887A+	0°C to +70°C	24 EDIP	DS12887A
DS12C887	0°C to +70°C	24 EDIP	DS12C887
DS12C887A	0°C to +70°C	24 EDIP	DS12C887
DS12C887A+	0°C to +70°C	24 EDIP	DS12C887

+Denotes a lead-free/RoHS-compliant device.

**A "+" anywhere on the top mark indicates a lead-free device, and an "N" indicates an industrial temperature range device.*

Chip Information

TRANSISTOR COUNT: 17,000

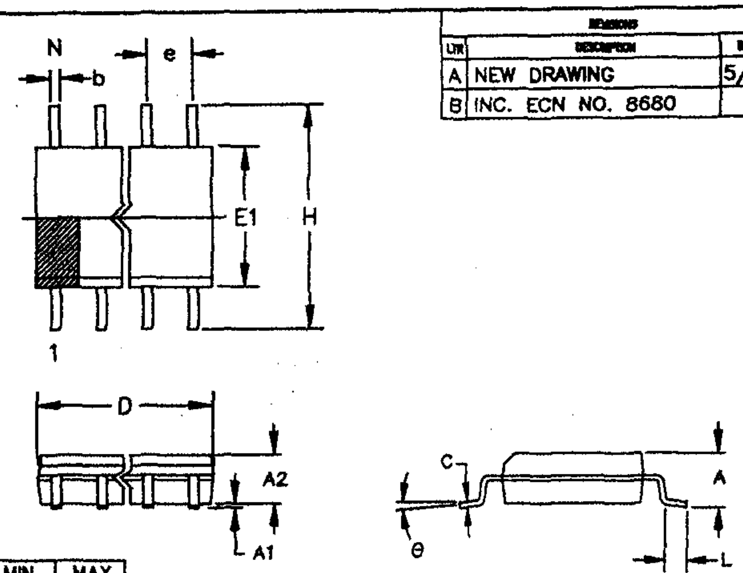
PROCESS: CMOS

SUBSTRATE CONNECTED TO GROUND

Package Information

For the latest package outline information, go to www.maxim-ic.com/DallasPackInfo.

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING	5/18	J.W.
B	INC. ECN NO. 8680		



LTR		MIN	MAX
A	IN.	0.094	0.105
	MM	2.39	2.67
A1	IN.	0.004	0.012
	MM	0.102	0.30
A2	IN.	0.089	0.095
	MM	2.26	2.41
b	IN.	0.013	0.020
	MM	0.33	0.51
C	IN.	0.009	0.013
	MM	0.229	0.33
D	IN.	——→	
	MM		
e	IN.	.050 BSC	
	MM	1.27 BSC	
E1	IN.	0.290	0.300
	MM	7.37	7.62
H	IN.	0.398	0.416
	MM	10.11	10.57
L	IN.	0.016	0.040
	MM	0.40	1.02
θ		0°	8°

16 PIN		18 PIN		20 PIN		24 PIN		28 PIN	
MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX
0.398	0.412	0.448	0.462	0.498	0.511	0.598	0.612	0.698	0.712
10.11	10.46	11.38	11.73	12.65	12.99	15.19	15.54	17.73	18.08

THE CHAMFER ON THE BODY IS OPTIONAL. IF IT IS NOT PRESENT, A TERMINAL 1 IDENTIFIER MUST BE POSITIONED SO THAT 1/2 OR MORE OF IT'S AREA IS CONTAINED IN THE HATCHED ZONE.

SIGNATURE		DATE
DOC. CONTROL:	J.WILKINS	5/94
ENGR. MGR:	B.W.MCARTY	5/94
MFG. ENGR:	C.M.SELLS	5/94
CHECKED BY:	C.M.SELLS	5/94
DRAWN BY:	M.W.C.	5/94
DO NOT SCALE DWG.		

DALLAS SEMICONDUCTOR MAXIM

TITLE: PACKAGE OUTLINE .300" SOIC 16,18,20,24&28 LD.

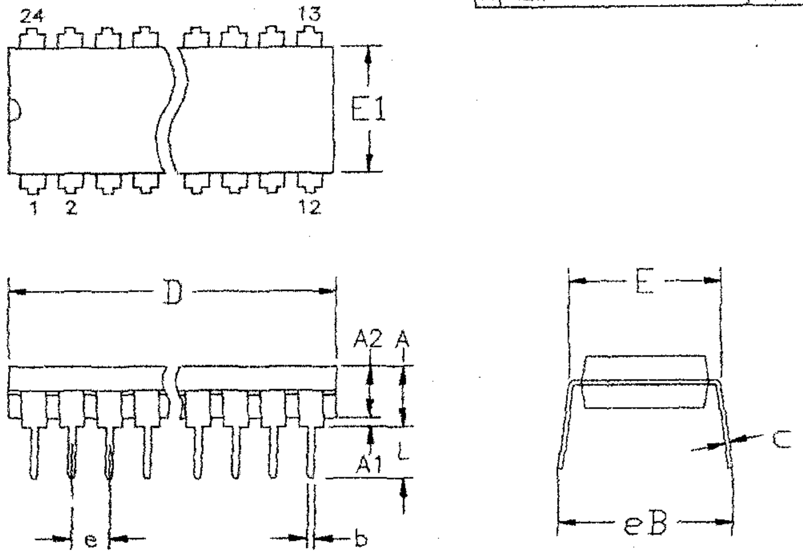
SIZE	FSCM NO	PART NO.	REV
A		56-G4009-001	B
SCALE N/A		SHEET 1 OF 1	

Package Information (continued)

or the latest package outline information, go to **www.maxim-ic.com/DallasPackInfo**.

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING	12/01	

56-G5000-003 REV A



	24 PIN	
	MIN	MAX
A	—	0.200
A1	0.015	—
A2	0.140	0.160
b	0.014	0.022
c	0.008	0.012
D	1.150	1.290
E	0.600	0.625
E1	0.530	0.555
e	0.090	0.110
L	0.115	0.145
eB	0.600	0.700

ALL DIMENSIONS ARE IN INCHES

SIGNATURE		DATE
DOC. CONTROL:		
ENGR. MGR:		
MFG. ENGR:		
CHECKED BY:	TWM	12/01
DRAWN BY:	JFD	12/01
DO NOT SCALE DWG.		

DALLAS SEMICONDUCTOR MAXIM

TITLE MARKETING OUTLINE, 24 LEAD PLASTIC DUAL-IN-LINE PACKAGE (0.600")

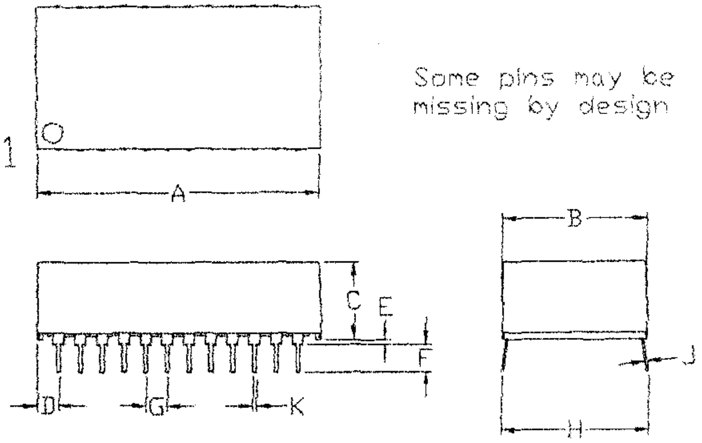
SIZE	FSCM NO	PART NO.	REV
A		56-G5000-003	A
SCALE N/A		SHEET 1 OF 1	

Package Information (continued)

For the latest package outline information, go to **www.maxim-ic.com/DallasPackInfo**.

56-G0001-001.EPS

REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	ECN 35439		



Some pins may be missing by design

ALL DIMENSIONS ARE IN INCHES. See page 2 for dimensions in mm

	24 PIN		28 PIN		32 PIN 740		40 PIN 720	
	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX
A	1.320	1.340	1.520	1.540	1.680	1.720	2.075	2.115
C	0.330	0.370	0.330	0.370	0.330	0.370	0.280	0.320
D	*0.100*	*0.130*	*0.100*	*0.130*	*0.070*	*0.100*	*0.070*	*0.100*

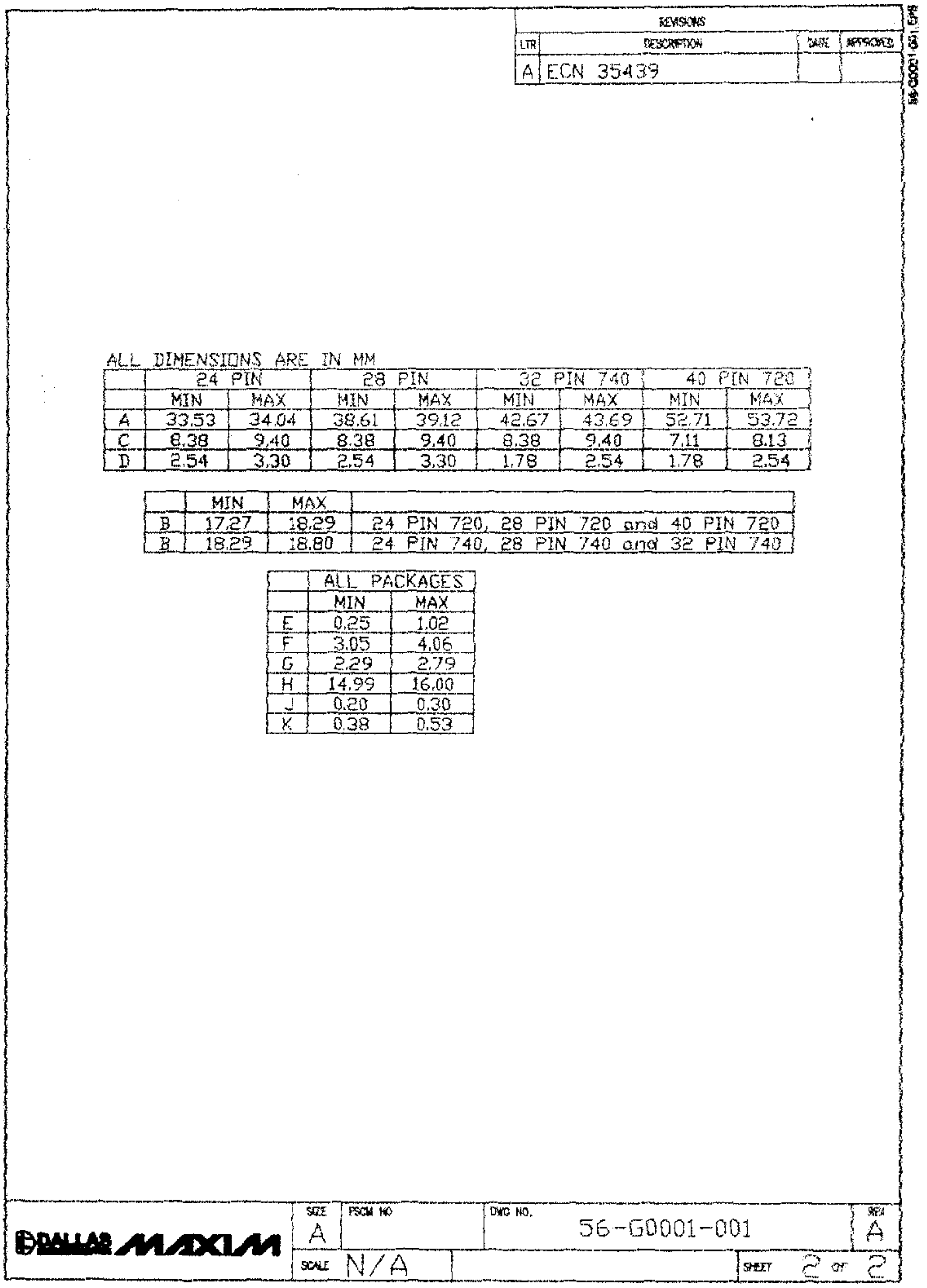
	MIN	MAX	
B	0.68	0.72	24 PIN 720, 28 PIN 720 and 40 PIN 720
B	*0.720*	*0.740*	24 PIN 740, 28 PIN 740 and 32 PIN 740

	ALL PACKAGES	
	MIN	MAX
E	0.010	0.040
F	0.120	0.160
G	0.090	0.110
H	0.590	0.630
J	0.008	0.012
K	0.015	0.021

SIGNATURE	DATE	DALLAS SEMICONDUCTOR MAXIM			
DOC. CONTROL:					
ENGR. MGR:		TITLE Marketing Outline Encapsulated DIP Package			
MFG. ENGR:					
CHECKED BY:		SIZE	FSCM NO	DWG NO. 56-G0001-001	REV
DRAWN BY: RMA	2/03	A			A
DIP_ModulesICSL.DDRW		SCALE N/A			SHEET 1 OF 2

Package Information (continued)

or the latest package outline information, go to **www.maxim-ic.com/DallasPackInfo**.



REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	ECN 35439		

56-G0001-001.EPS

ALL DIMENSIONS ARE IN MM

	24 PIN		28 PIN		32 PIN 740		40 PIN 720	
	MIN	MAX	MIN	MAX	MIN	MAX	MIN	MAX
A	33.53	34.04	38.61	39.12	42.67	43.69	52.71	53.72
C	8.38	9.40	8.38	9.40	8.38	9.40	7.11	8.13
D	2.54	3.30	2.54	3.30	1.78	2.54	1.78	2.54

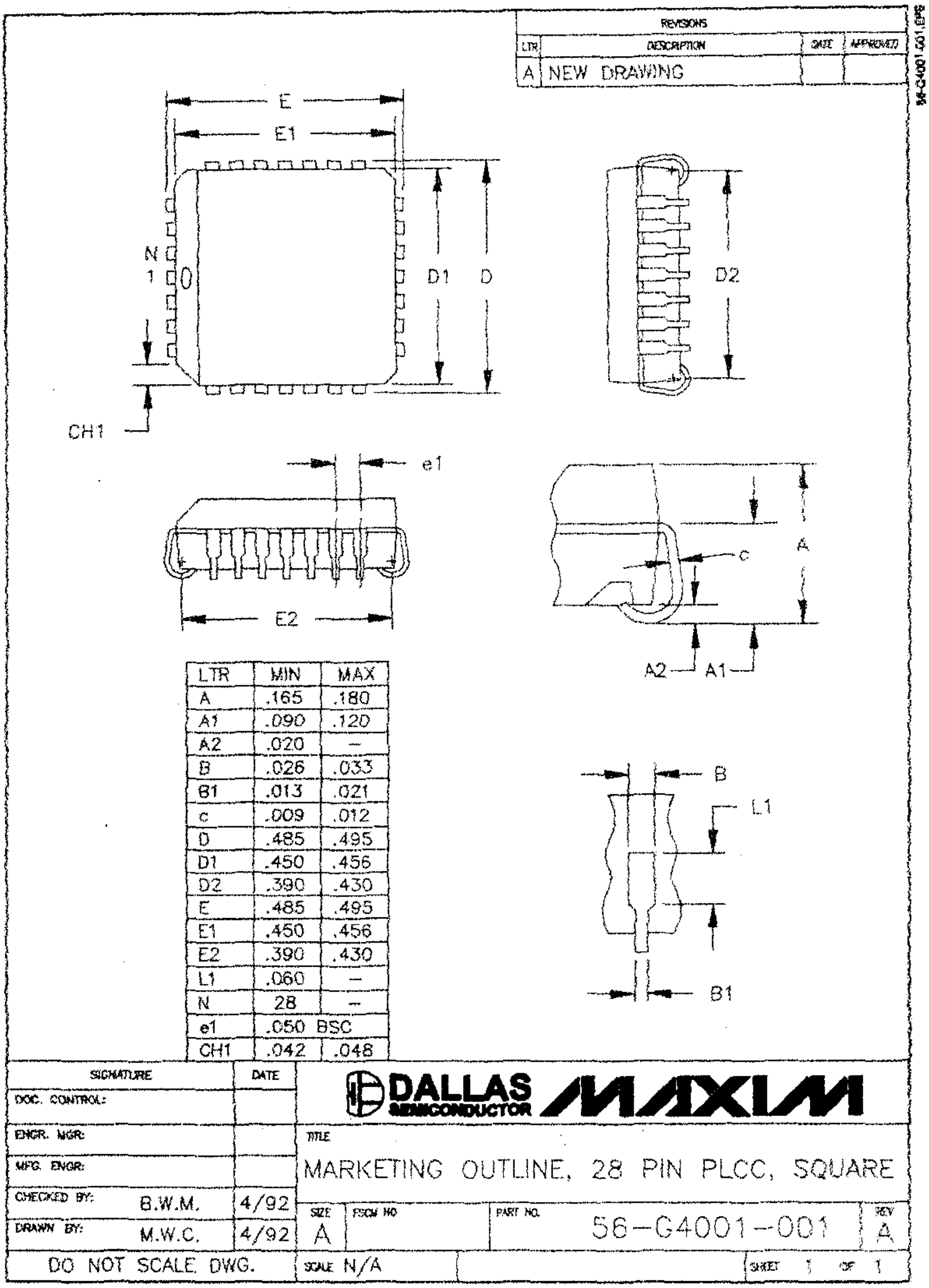
	MIN	MAX	
B	17.27	18.29	24 PIN 720, 28 PIN 720 and 40 PIN 720
B	18.29	18.80	24 PIN 740, 28 PIN 740 and 32 PIN 740

	ALL PACKAGES	
	MIN	MAX
E	0.25	1.02
F	3.05	4.06
G	2.29	2.79
H	14.99	16.00
J	0.20	0.30
K	0.38	0.53

DALLAS MAXIM	SIZE A	FSCM NO	DWG NO. 56-G0001-001	REV A
	SCALE N/A			SHEET 2 OF 2

Package Information (continued)

For the latest package outline information, go to www.maxim-ic.com/DallasPackInfo.



REVISIONS			
LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING		

LTR	MIN	MAX
A	.165	.180
A1	.090	.120
A2	.020	–
B	.026	.033
B1	.013	.021
c	.009	.012
D	.485	.495
D1	.450	.456
D2	.390	.430
E	.485	.495
E1	.450	.456
E2	.390	.430
L1	.060	–
N	28	–
e1	.050 BSC	
CH1	.042	.048

SIGNATURE		DATE
DOC. CONTROL:		
ENGR. MGR:		
MFG. ENGR:		
CHECKED BY:	B.W.M.	4/92
DRAWN BY:	M.W.C.	4/92
DO NOT SCALE DWG.		

DALLAS SEMICONDUCTOR MAXIM

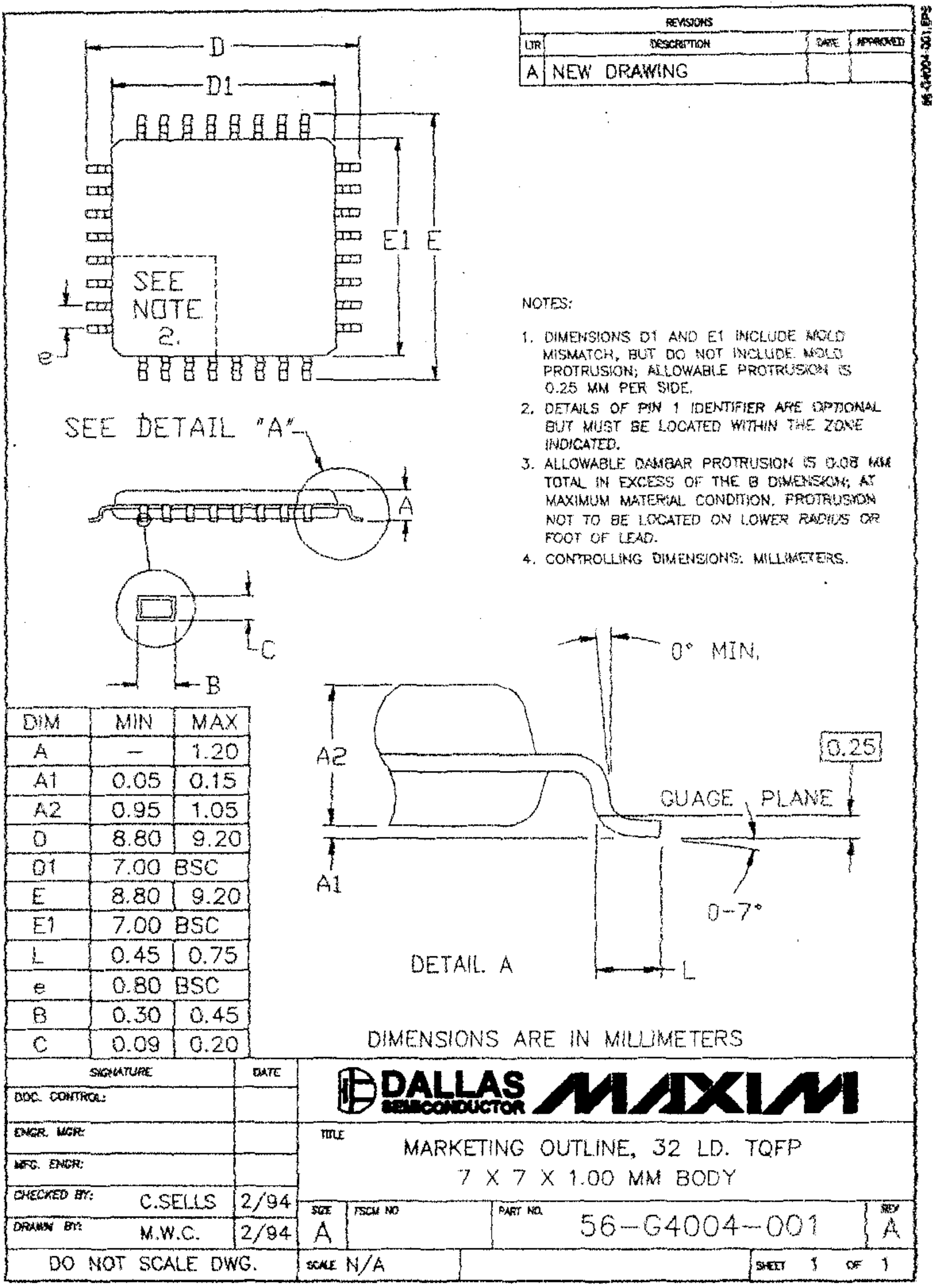
TITLE
MARKETING OUTLINE, 28 PIN PLCC, SQUARE

SIZE	FSCM NO	PART NO.	REV
A		56-G4001-001	A

SCALE N/A SHEET 1 OF 1

Package Information (continued)

or the latest package outline information, go to **www.maxim-ic.com/DallasPackInfo**.



REVISIONS

LTR	DESCRIPTION	DATE	APPROVED
A	NEW DRAWING		

NOTES:

1. DIMENSIONS D1 AND E1 INCLUDE MOLD MISMATCH, BUT DO NOT INCLUDE MOLD PROTRUSION; ALLOWABLE PROTRUSION IS 0.25 MM PER SIDE.
2. DETAILS OF PIN 1 IDENTIFIER ARE OPTIONAL BUT MUST BE LOCATED WITHIN THE ZONE INDICATED.
3. ALLOWABLE DAMBAR PROTRUSION IS 0.08 MM TOTAL IN EXCESS OF THE B DIMENSION; AT MAXIMUM MATERIAL CONDITION. PROTRUSION NOT TO BE LOCATED ON LOWER RADIUS OR FOOT OF LEAD.
4. CONTROLLING DIMENSIONS: MILLIMETERS.

DIM	MIN	MAX
A	—	1.20
A1	0.05	0.15
A2	0.95	1.05
D	8.80	9.20
D1	7.00 BSC	
E	8.80	9.20
E1	7.00 BSC	
L	0.45	0.75
e	0.80 BSC	
B	0.30	0.45
C	0.09	0.20

DIMENSIONS ARE IN MILLIMETERS

SIGNATURE		DATE
DOC. CONTROL:		
ENGR. MGR:		
MFG. ENGR:		
CHECKED BY:	C.SELLS	2/94
DRAWN BY:	M.W.C.	2/94
DO NOT SCALE DWG.		

DALLAS SEMICONDUCTOR MAXIM

TITLE MARKETING OUTLINE, 32 LD. TQFP 7 X 7 X 1.00 MM BODY

SIZE	FSCM NO	PART NO.	REV
A		56-G4004-001	A
SCALE N/A		SHEET 1 OF 1	

xim cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim product. No circuit patent licenses are plied. Maxim reserves the right to change the circuitry and specifications without notice at any time.

BIODATA

Nama	: Sunoto Pardi
NRP	: 5103002023
Tempat/Tanggal Lahir	: Surabaya / 10 Nopember 1983
Alamat	: Jl. Geteng Besar 22 Surabaya

Riwayat Pendidikan:

- Tahun 1996 Lulus SD Ketabang Kali Surabaya
- Tahun 1999 Lulus SLTPK Gloria Surabaya
- Tahun 2002 Lulus SMU IMKA Surabaya
- Tahun 2007 Lulus Sarjana Jurusan Teknik Elektro Fakultas Teknik Universitas Katolik Widya Mandala Surabaya