LAMPIRAN A

Foto Alat dan Skematik Rangkaian

Foto Alat A-1
Skematik Rangkaian Master A-2
Skematik Rangkaian Slave A-3
LAMPIRAN B

Listing Program Master
;traffic master source code

.include "J:\AVR\m8535def.inc"
def tmp=r24
def txbyte=r25
def rxbyte=r18

equ fclock=11059200
equ baud_rate=1200 ;2400
equ ubbr_value=(fclock/(16*baud_rate))-1
equ tmr1_value=0xd5d0 ;0xeae8 : set waktu tmr1 result=1detik

equ flag1=$60 ; anime *
equ DTtmp1=$61
equ cmenu=$62
equ flag2=$63 ; refresh menu
equ flag3=$64 ; lock tbl menu

equ colour1=$65 ; green=0, orange=1, red=2
equ colour2=$66 ; green=0, orange=1, red=2
equ colour3=$67 ; green=0, orange=1, red=2
equ colour4=$68 ; green=0, orange=1, red=2

equ status1=$69 ; sepi=1, sedang=10, padat=00
equ status2=$6a ; sepi=1, sedang=10, padat=00
equ status3=$6b ; sepi=1, sedang=10, padat=00
equ status4=$6c ; sepi=1, sedang=10, padat=00

equ gilir=$6d ; A(master)=0, B=1, C=2, D=3

equ REDs=$6e
equ REDm=$6f
equ REDl=$70
.equ GREENs=$71
.equ GREENm=$72
.equ GREENl=$73

equ flag4=$74 ; lock tbl UPSmall

equ flag5=$75 ; lock tbl UPMedium

equ flag6=$76 ; lock tbl UPLarge

equ flag7=$77 ; update traffic

equ trafic1=$78 ; green=100, orange=010, red=001
equ trafic2=$79 ; green=100, orange=010, red=001
equ trafic3=$7a ; green=100, orange=010, red=001
equ trafic4=$7b ; green=100, orange=010, red=001

equ REDrun=$7c
.equ GREENrun=$7d

equ dtRUNa=$7e

equ stepRa=$7f ; status simpangan saat run? dieksekusi/giliran, sepi=00, sedang=01, padat=10

equ REDrunA=$80
.equ REDrunB=$81
.equ REDrunC=$82
.equ REDrunD=$83
.equ GREENrunA=$84
.equ GREENrunB=$85
.equ GREENrunC=$86
.equ GREENrunD=$87
.equ dtRUNb=$88
.equ dtRUNc=$89
.equ dtRUNd=$8a
.equ stepRb=$8b
.equ stepRc=$8c
.equ stepRd=$8d
.equ stepRc=$8e
.equ flagG1=$90
.equ flagG2=$91
.equ flagG3=$92
.equ flagG4=$93
.equ flagR1=$94
.equ flagR2=$95
.equ flagR3=$96
.equ flagR4=$98
.equ flagR3b=$99
.equ flagR3c=$9a
.equ flagR3d=$9b
.equ dataSER=$9c
.equ cirim=$9d
.equ flgsend=$9e
.equ dtUSER=$9f
.equ RXcpl=0a0
.equ flagGC1=0a1
.equ flagGCx=0a2
.equ flagD1=0a3
.equ flagD2=0a4
.equ bufPC=0a5
.equ kail=0a6
.equ randomCD=0a7 ;random code:1-5
.equ dataTX=0a8 ;test
.equ TranCD=0a9 ;transmit randomCD
.equ flagR5b=0b0
.equ flagR5c=0b1
.equ flagR5d=0b2
.equ code2=0b3
.equ code3=0b4
.equ code4=0b5
.equ status2x=0b6
.equ status3x=0b7
.equ status4x=0b8
.equ DTTest1=0b9
.equ DTTest2=0ba
.equ flag8=0bb ;resend data Tx
.equ bilR1=0bc
.equ bilR2=0bd
.equ flagC2=0be
.equ flagC3=0bf
.equ flagC4=0c0

.org 0x0000
rjmp init

.org 0x0008 ;vector int tmr1 overflow
rjmp Ri_TMR1

.org 0x000b
rjmp usart_rxc

init:
    ldi r16,low(ramend)
    out spl,r16
    ldi r16,high(ramend)
    out sph,r16

;---
;---setting fungsi port
    ldi r16,0xff
    ldi r19,0x00
    ldi r18,0b00011111
    out ddrA,r16 ;set port A sebagai output
    out portA,r16
    out ddrB,r16 ;set port B sebagai output
    out portB,r16

B - 2
out ddrC,r19 ;set port C sebagai masukan pull up
out portC,r16
out ddrD,r16 ;set port D sebagai masukan pull up
out portD,r16
sts flag2,r16 ;refresh menu
sts flag1,r19
ldi r16,1
sts cmenu,r16 ;def=menu1
sts dataTXx,r19 ;test
:
ldi r16,15 ;small red
sts REDs,r16
ldi r16,10 ;medium red
sts REDm,r16
ldi r16,5 ;large red
sts REDl,r16
:
ldi r16,5 ;small green
sts GREENs,r16
ldi r16,10 ;medium green
sts GREENm,r16
ldi r16,15 ;large green
sts GREENl,r16
:
ldi r16,0b11000000 ;0xff
sts status1,r16
sts status2,r16
sts status3,r16
sts status4,r16
ldi r16,0xff
sts flagG1,r16 ;aktifkan timing master pertama kali
sts dataSER,r16
:
ldi r16,0x0
sts bilR1,r16
sts flagC2,r16
sts flagC3,r16
sts flagC4,r16
ldi r16,0x1
sts bilR2,r16
:
ldi r16,0
ldi r17,0xb3
sts flag8,r16
sts DTest1,r16
sts DTest2,r16
sts flagR5b,r16
sts flagR5c,r16
sts flagR5d,r16
sts TranCD,r16
sts kali,r16
sts bufPC,r16
sts flagD1,r16
sts flagD2,r16
sts flagGC1,r16 ;gilir client diaktifkan setelah 1x loop
sts flagGCx,r16
sts dtUSER,r17
sts RXcplt,r16
sts ckrim,r16
sts flagSEND,r16
sts gilir,r16 ;gilir default
sts dtRUNa,r16
sts dtRUNb,r16
sts dtRUNc,r16
sts dtRUNd,r16
sts stepRa,r16
sts stepRb,r16
sts stepRc,r16
sts stepRd,r16
sts flagG2,r16
sts flagG3.r16
sts flagG4.r16
sts flagR1.r16
sts flagR2.r16
sts flagR3.r16
sts flagR3b.r16
sts flagR3c.r16
sts flagR3d.r16
ldi r16,0x00000001 ;set all default to RED
sts tрафic1.r16
sts tрафic2.r16
sts tрафic3.r16
sts tрафic4.r16
sts randomCD.r16 ;range 1-5

---end setting port

;---init register
rcall initLCD
rcall initTimer1
rcall initUSART
; rcall setting
sts DTmp1.r19
rcall tmenu0
rcall delay1000m
rcall setTa
rcall settingEN
sbi portD,2 ;disable TX
; cbи portD,2 ;enable TX

;==================================
UTAMA:
call grupT ;grup tombol
call menu1 ;manager menu
call anim1 ;animasi *
call grupDCP ;test
call calcm ;calculasi timing трафic
call updateLED
call kirimGC
call grupTX
rjmp UTAMA

;=====================================

;decode status limit
;decode register "status2";"status3" dan "status4", dengan masing2x код acaknya

deоcript grup
grupDCP:
call deоryptDTc2
call deоryptDTc3
call deоryptDTc4
ret

deоrypt data client1
deоryptDTc2q:
ret
deоryptDTc2:
lds r16,flagC2
cpi r16,0xff
brne deоryptDTc2q
clr r16
sts flagC2.r16
/
lds r18,code2
call setTabел
lds r16,status2
call run_deоrypt
call status2x.r16
ret
; descrypt data client2
decrpDTc3q:
  ret
de crpDTc3:
  lds r16,flagC3
  cpi r16,0xff
  brne decrpDTc3q
  clr r16
  sts flagC3,r16
  
  lds r18,code3
  rcall setTabel
  lds r16,status3
  rcall run_dcript
  sts status3x,r16
  ret

; descrypt data client3
decrpDTc4q:
  ret
de crpDTc4:
  lds r16,flagC4
  cpi r16,0xff
  brne decrpDTc4q
  clr r16
  sts flagC4,r16
  
  lds r18,code4
  rcall setTabel
  lds r16,status4
  rcall run_dcript
  sts status4x,r16
  ret

; encrypt data
encrDT:
  lds r18,randomCD
  sts TranCD,r18
  rcall setTabel
  rcall run_cript
  ret

; load encrypt code, test
setTabel:
  ldi zl,low(2*criptCode)
  ldi zh,high(2*criptCode)
  ret

; load encrypt table, test
run_cript:
  ; tentukan baris tabel yg akan diakses
  dec r18
  cpi r18,0
  breq goCrip
  addw Zh,zl,1 ; cek nilai register "randomCD"
  jmp run_cript
goCrip:
  lpm
  mov r17,r0
  lds r16,dataTX
  eor r16,r17
  sts dataTXx,r16
  ret

; load decrupt table, test
run_dcript:
dec r18
  cpi r18,0
  breq godCrip
  adiw Zh:Zl,1
  jmp run_dcript ;cek nilai register "randomCD"

godCrip:
  lpm
  mov r17, r0
  eor r16, r17
  andi r16, 0b11000000
  ret

grupTXq:
  sbi portD, 2  ;disable TX
  ret

grupTX:
  lds r16, flgSend
  cpi r16, 0xff
  bne grupTXq
  ;/
  lds r16, flag8  ;resend tiap 1 detik
  cpi r16, 0xff
  bne grupTXq
  clr r16
  sts flag8, r16
  rcall setingDIS
  <
  cbi portD, 2  ;enable TX
  >
  rcall kirimC1  ;payload
  rcall kirimC1x  ;payload
  <
  rcall kirimC2
  rcall kirimC2x
  <
  rcall kirimC3
  rcall kirimC3x
  <
  rcall setingEN
  ret

;kirim ke client1/b
kirimC1:
  lds r16, trafic2
  andi r16, 0b01100111
  sts dataTX, r16
  rcall encrDT ;test encript?
  rcall kirimDum  ;dummy send
  nop
  nop
  nop
  ldi txbyte, 0x8a  ;r16
  ;header1 from master
  rcall usart Tx
  nop
  nop
  nop
  ldi txbyte, 0x9b  ;r16
  rcall usart Tx
  nop
  nop
  nop
  ldi txbyte, 0xdb  ;r16
  rcall usart Tx
  nop
  nop
  lds txbyte, dataTXx
rcall usart_tx
nop
nop
nop
lds r16,TranCD ;test?
sts DTTest2,r16 ;test?
lds txbyte,TranCD ;code random, test
rcall usart_tx
nop
nop
nop
ret

;@kirim ke client2/c
kirimC2:

lds r16,trafic3
andi r16,0b01100111
sts dataTX,r16
rcall encrDT ;test encrpt

kirimC2x:

call kirimDum ;dummy send
nop
nop
nop
ldi txbyte,0x8a ;r16 ;header1
rcall usart_tx
nop
nop
nop
ldi txbyte,0x9b ;r16
rcall usart_tx
nop
nop
nop
ldi txbyte,0xdc ;r16 ;user code:#-b,#-c,#-d
rcall usart_tx
nop
nop
nop
lds txbyte,dataTXx
rcall usart_tx
nop
nop
nop
lds txbyte,TranCD ;code random, test
rcall usart_tx
nop
nop
nop
ret

;@kirim ke client3/d
kirimC3:

lds r16,trafic4
andi r16,0b01100111
sts dataTX,r16
rcall encrDT ;test encrpt

kirimC3x:

call kirimDum ;dummy send
nop
nop
nop
ldi txbyte,0x8a ;r16 ;header1
rcall usart_tx
nop
nop
nop
ldi txbyte,0x9b ;r16
rcall usart_tx
nop
nop
nop
ldi txbyte,0x8a
rcall usart_tx

B - 7
nop
nop
nop
ldi txbyte,0xdd
rcall usart_tx
nop
nop
nop
lds txbyte,dataTXx
rcall usart_tx
nop
nop
nop
ldi txbyte,TranCD
rcall usart_tx
nop
nop
nop
rcall kirimDum
ret

;@kirim code gilir client
kirimGCq:
  ;kondisi enable TX ngikut grupTx
  ret
kirimGC:
  lds r16,flagGCx
  cpi r16,0xff
  brne kirimGCq
  ldi r16,0
  sts flagGCx,r16
  rcall setingDIS
  <
  cbi portD,2
  ;enable TX
  nop
  nop
  nop
  rcall kirimDum
  ;dummy send
  nop
  nop
  nop
  rcall payload
  rcall payload
  rcall setingEN
  ret

payload:
  ldi txbyte,0x8a
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0x9b
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0xe6
  rcall usart_tx
  nop
  nop
  nop
  ; lds txbyte,dataTX
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop

  ;
@kirim dummy data
kirimDum:
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  ret

@test poin
testP1:
  ldi r20,0x85
  rcall kirim_add
  lds r18,DTTest1
  ;DTtest1  ;randomCD  ;dtUSER
  rcall digi2
  :
  ldi r20,0x87
  rcall kirim_add
  lds r18,DTTest2
  ;randomCD  ;flagG2  ;stepRg
  rcall digi2
  ret

@update trafic LED
updateLED:
  lds r17,trafic1
  rcall trig574
  ret

;aktifkan sekali code ganti giliran client, sebelum master betul2x off
gilirFLAGq:
  ret
gilirFLAG:
  lds r16,flagGC1
  cpi r16,0xff
  brne gilirFLAGq
  ldi r16,0
  sts flagGC1,r16
  >
  ldi r16,0xff
  sts flagGCx,r16
  ret

;calculate trafic timing
calcmQ:
  ret
calcm:
  lds r16,flag7
  cpi r16,0xff
  brne calcmQ
  ;cek timing update?
  ldi r16,0
sts flag7,r16
;
rcall GOMaster
rcall GOCient1
call GOCient2
call GOCient3
ret
;

GOMasterQ:
    ret
GOMaster:
    lds r16,flagG1
cpi r16,0xff
brne GOMasterQ
    ;master

GHCJ1:
    lds r16,stepRa
cpi r16,0
brne gRG1
    ;giliran hijau?
    /rcall readSM
    ;baca status limit master saat ijo saja
    ;;ldi r16,0x00
    ;disable all TX
sts flgSend,r16
call gilirFLAG
    ;aktifkan sekali
    /
ldsr16,dtonNa
ldsr17,GREENrunA
cp r16,r17
brlo hslA1
    ;reset & go to next step
ldi r16,0
sts dtonNa,r16
ldi r16,0b00000010
    ;orange
sts trafic1,r16
ldi r16,1
    ;naik ke step orange
sts stepRa,r16
ret

hslA1:
    ldi r16,0b00000100
    ;hijau
sts trafic1,r16
ldsr16,dtonNa
inc r16
sts dtonNa,r16
ret

    ;>>>

GGRG1:
    cpi r16,1
    ;giliran orange?
brmegrD1
    /
ldsr16,dtonNa
ldi r17,3
    ;orange nyala 3detik saja
cp r16,r17
brlo hslA2
    ;reset & go to next step
ldi r16,0
sts dtonNa,r16
ldi r16,0b00000001
    ;red
sts trafic1,r16
ldi r16,2
    ;naik ke step red
sts stepRa,r16
ldi r16,0
sts stepRb,r16
    ;aktifkan next polling
call setTb
    ;set timing client1
ldi r16,0xff
sts flgG2,r16
ret

hslA2:
ldi r16,0b00000010 ;orange
sts traffic1,r16
lds r16,dtRUNa ;inc dtRUN
inc r16
sts dtRUNa,r16
ret

>>> gRD1:

/*
 lds r16,dtRUNa
 lds r17,REDrunA
 cp r16,r17
 brlo hslA3
 :reset & go to next step
 ldi r16,0
 sts dtRUNa,r16
 ldi r16,0b00000001 ;red
 sts traffic1,r16
 ldi r16,0 ;reset ke giliran hijau utk next client
 sts stepRa,r16
 :disable GOmaster
 ldi r16,0
 sts flagG1,r16
 ret

hslA3:

ldi r16,0b00000001 ;red
sts traffic1,r16
lds r16,dtRUNa ;inc dtRUN
inc r16
sts dtRUNa,r16
ret

<<<<>
GOclient1Q:
ret
GOclient1:

lds r18,flagG2
cpi r18,0xff
brne GOclient1Q
>
ldi r16,0xff ;back enable TX
sts flagSend,r16
ldi r16,0xff
sts flagGC1,r16
:<

gHJ2:

lds r18,stepRb
cpi r18,0 ;giliran hijau?
brne gRG2
/>

lds r18,dtRUNb
lds r19,GREENrunB
cp r18,r19
brlo hslB1
:reset & go to next step
ldi r18,0
sts dtRUNb,r18
ldi r18,0b000000010 ;orange
sts traffic2,r18
ldi r18,1 ;naik ke step orange
sts stepRb,r18
ret

hslB1:

ldi r18,0b000000100 ;hijau
sts traffic2,r18
lds r18,dtRUNb
inc r18
sts dtRUNb,r18
ret

B - 11
gRG2:

```assembly
cpi r18,1 ;giliran orange?
brne gRD2
/ lds r18,dtRUNb ldi r19,3 ;orange nyala 3detik saja
cp r18,r19 brlo hslB2 ;reset & go to next step
ldi r18,0 sts dtRUNb,r18 ldi r18,0b00000001 ;red sts trafic2,r18 ldi r18,2 ;naik ke step red
sts stepRb,r18 ldi r18,0 sts stepRc,r18 ;aktifkan next polling
rcall setTc ;set timing client1
ldi r18,0xff sts flagG3,r18
ret
```

hslB2:

```assembly
ldi r18,0b000000010 ;orange sts trafic2,r18 lds r18,dtRUNb ;inc dtRUN inc r18 sts dtRUNb,r18 ret
```

>>>

```
gRD2:
```

```assembly
/ lds r18,dtRUNb lds r19,REDrunB
cp r18,r19 brlo hslB3 ;reset & go to next step
ldi r18,0 sts dtRUNb,r18 ldi r18,0b00000001 ;red sts trafic2,r18
ldi r18,0 ;reset ke giliran hijau utk next client
sts stepRb,r18 ;disable GO client1
ldi r18,0
sts flagG2,r18
ret
```

hslB3:

```assembly
ldi r18,0b00000001 ;red sts trafic2,r18 lds r18,dtRUNb ;inc dtRUN inc r18
sts dtRUNb,r18 ret
```

<<<<<<<<<<<

GOclient2Q:

```assembly
ret
```

GOclient2:

```assembly
lds r20,flagG3 cpi r20,0xff brne GOclient2Q
```

ghJ3:

```assembly
lds r20,stepRc
cpi r20,0 ;giliran hijau?
brne gRG3 /
lds r20,dtRUNc
```
lds r21, GREENrunC
cp r20, r21
brlo hsiC1
;reset & go to next step
ldi r20, 0
sts dtRUNC, r20
ldi r20, 0b00000010 ;orange
sts trafic3, r20
ldi r20, 1 ;naik ke step orange
sts stepRc, r20
ret

hsiC1:
ldi r20, 0b00000010 ;hijau
sts trafic3, r20
lds r20, dtRUNC
inc r20
sts dtRUNC, r20
ret

ghRG3:
cpi r20, 1 ;giliran orange?
brne gRD3
; lds r20, dtRUNC
ldi r21, 3 ;orange nyala 3detik saja
cp r20, r21
brlo hsiC2
;reset & go to next step
ldi r20, 0
sts dtRUNC, r20
ldi r20, 0b00000001 ;red
sts trafic3, r20
ldi r20, 2 ;naik ke step red
sts stepRc, r20
ldi r20, 0
sts stepRd, r20
;aktifkan next polling
rcall setTd
; set timing client1
ldi r20, 0xff
sts flagG4, r20
ret

hsiC2:
ldi r20, 0b00000010 ;orange
sts trafic3, r20
lds r20, dtRUNC ;inc dtRUN
inc r20
sts dtRUNC, r20
ret

ghRD3:
; lds r20, dtRUNC
lds r21, REDrunC
cp r20, r21
brlo hsiC3
;reset & go to next step
ldi r20, 0
sts dtRUNC, r20
ldi r20, 0b00000001 ;red
sts trafic3, r20
ldi r20, 0 ;reset ke giliran hijau utk next client
sts stepRc, r20
;dioble GOclient2
ldi r20, 0
sts flagG3, r20
ret

hsiC3:
ldi r20, 0b00000001 ;red
sts trafic3, r20
lds r20, dtRUNC ;inc dtRUN

inc r20
sts dtRUNc,r20
ret

<<<>>>  
GOclient3Q:
    ret
GOclient3:
    lds r22,flagG4
    cpi r22,0xff
    brne GOclient3Q

  gHJ4:
    lds r22,stepRd
    cpi r22,0 ;giliran hijau?
    brne gRG4
    lds r22,dtRUNd
    lds r23,GREENrunD
    cp r22,r23
    brlo hsiD1 ;reset & go to next step
    ldi r22,0
    sts dtRUNd,r22
    ldi r22,0b00000010 ;orange
    sts trafic4,r22
    ldi r22,1 ;naik ke step orange
    sts stepRd,r22
    ret

  hsiD1:
    ldi r22,0b000000100 ;hijau
    sts trafic4,r22
    lds r22,dtRUNd
    inc r22
    sts dtRUNd,r22
    ret

  >>>
  gRG4:
    cpi r22,1 ;giliran orange?
    brne gRD4
    lds r22,dtRUNd
    ldi r23,3 ;orange nyala 3detik saja
    cp r22,r23
    brlo hsiD2 ;reset & go to next step
    ldi r22,0
    sts dtRUNd,r22
    ldi r22,0b00000001 ;red
    sts trafic4,r22
    ldi r22,2 ;naik ke step red
    sts stepRd,r22
    ldi r22,0
    sts stepRa,r22
    ;aktifkan next polling
    rcall setTa ;set timing client1
    ldi r22,0xff
    sts flagG1,r22
    ret

  hsiD2:
    ldi r22,0b000000010 ;orange
    sts trafic4,r22
    lds r22,dtRUNd ;inc dtRUN
    inc r22
    sts dtRUNd,r22
    ret

  >>>>
  gRD4:
    

lds r22,dtRUNd
lds r23,REDrunD
cp r22,r23
brlo hslD3
:reset & go to next step
ldi r22,0
sts dtRUNd,r22
ldi r22,0b00000001 ;red
sts traffic4,r22
ldi r22,0 ;reset ke giliran hijau utk next client
sts stepRd,r22
:disable GOclient3
ldi r22,0
sts flagG4,r22
ret

hslD3:
ldi r22,0b00000001 ;red
sts traffic4,r22
lds r22,dtRUNd ;inc dtRUN
inc r22
sts dtRUNd,r22
ret

#:set timing run

>>>S
TsmallA:
lds r16,REDS
sts REDrunA,r16
lds r16,GREENs
sts GREENrunA,r16
ret

TsmallB:
lds r16,REDS
sts REDrunB,r16
lds r16,GREENs
sts GREENrunB,r16
ret

TsmallC:
lds r16,REDS
sts REDrunC,r16
lds r16,GREENs
sts GREENrunC,r16
ret

TsmallD:
lds r16,REDS
sts REDrunD,r16
lds r16,GREENs
sts GREENrunD,r16
ret

>>>M
TmediumA:
lds r16,REDm
sts REDrunA,r16
lds r16,GREENm
sts GREENrunA,r16
ret

TmediumB:
lds r16,REDm
sts REDrunB,r16
lds r16,GREENm
sts GREENrunB,r16
ret
TmediumC:
    lds r16, REDm
    sts REDrunC, r16
    lds r16, GREENm
    sts GREENrunC, r16
    ret

TmediumD:
    lds r16, REDm
    sts REDrunD, r16
    lds r16, GREENm
    sts GREENrunD, r16
    ret

>>>L
TlargeA:
    lds r16, REDl
    sts REDrunA, r16
    lds r16, GREENl
    sts GREENrunA, r16
    ret

TlargeB:
    lds r16, REDl
    sts REDrunB, r16
    lds r16, GREENl
    sts GREENrunB, r16
    ret

TlargeC:
    lds r16, REDl
    sts REDrunC, r16
    lds r16, GREENl
    sts GREENrunC, r16
    ret

TlargeD:
    lds r16, REDl
    sts REDrunD, r16
    lds r16, GREENl
    sts GREENrunD, r16
    ret

;@baca status limit master
readSM:
    in r16, pinC
    andi r16, 0b11000000
    sts bufPC, r16
    cpi r16, 0b00000000 ;limit1-2 ketekan
    breq cekv2
    brne cekv2
    cpi r16, 0xff
    breq cekNOq
    clr r16
    sts kali, r16
    sts flagD2, r16
    ser r16
    sts flagD1, r16
    ldi r16, 0b11000000
    sts status1, r16
    ret

cekv2:
    cpi r16, 0b10000000 ;hanya limit1 ketekan
    brne cekv3
    lds r16, flagD2
    cpi r16, 0xff
    breq cekNOq
clr r16
sts kali,r16
sts flagD1,r16
ser r16
sts flagD2,r16
ldi r16,0b11000000
sts status1,r16
ret

cekv3:
cpi r16,0b01000000
;hanya limit2 ketekan=abaikan
brne cekNO
ldi r16,0b11000000
sts bufPC,r16
clr r16
sts flagD1,r16
sts flagD2,r16
ret

cekNO:
sts status1,r16
clr r16
sts flagD1,r16
sts flagD2,r16
ret

cekNOq:
ret

ret

;<tampilkan hasil cek trafic
;master LED
hasilLED:
ldi r20,0x83
rcall kirim_add
lds r16,traffic1
cpi r16,0b00000001
brne has2LED
ldi r20,'R'
rcall kirim_asci
ret

has2LED:
cpi r16,0b00000010
brne has3LED
ldi r20,'O'
rcall kirim_asci
ret

has3LED:
ldi r20,'G'
rcall kirim_asci
ret

;master kapasitas
hasil:
ldi r20,0x82
rcall kirim_add
lds r16,status1
cpi r16,0b00000000
brne has2
ldi r20,'L'
rcall kirim_asci
ret

has2:
cpi r16,0b10000000
brne has3
ldi r20,'M'
rcall kirim_asci
ret

has3:
ldi r20,0b00000000
rcall kirim_asci
ret

;;>>master set timing def
setTa:
    lds r16, status1
    cpi r16, 0b00000000
    brne sHasA1
    rcall TlargeA
    ret
sHasA1:
    cpi r16, 0b10000000
    brne sHasA2
    rcall TmediumA
    ret
sHasA2:
    :selebihnya dianggap small
    rcall TsmallA
    ret

;>>>>>>>>>
;client1-a LED
hasilc1LED:
    ldi r20, 0x8d
    rcall kirim_add
    lds r16, trafic2
    cpi r16, 0b00000001
    brne has2c1LED
    ldi r20, 'R'
    rcall kirim_asci
    ret
has2c1LED:
    cpi r16, 0b00000010
    brne has3c1LED
    ldi r20, 'O'
    rcall kirim_asci
    ret
has3c1LED:
    ldi r20, 'G'
    rcall kirim_asci
    ret

;client1-a
hasilc1:
    ldi r20, 0x8c
    rcall kirim_add
    lds r16, status2x
    cpi r16, 0b00000000
    brne has3c1
    ldi r20, 'L'
    rcall kirim_asci
    ret
has3c1:
    cpi r16, 0b10000000
    brne has3c1
    ldi r20, 'M'
    rcall kirim_asci
    ret
has3c1:
    : cpi r16, 0b00000000
    ldi r20, 'S'
    rcall kirim_asci
    ret

;>>client1 set timing def
setTb:
    lds r16, status2x
    cpi r16, 0b00000000
    brne sHasB1
    rcall TlargeB
    ret
sHasB1:
    cpi r16, 0b10000000
    brne sHasB2
    rcall TmediumB
    ret
sHasB2:
  ;selebihnya dianggap small
  rcall TamallB
  ret

;;;;;;
;client2: b LED
hasil2LED:
  ldi r20,0xc6
  rcall kirim_add
  lds r16,traffic3
  cpi r16,0b00000001
  brne has2c2LED
  ldi r20,'R'
  rcall kirim_asci
  ret
has2c2LED:
  cpi r16,0b00000010
  brne has3c2LED
  ldi r20,'O'
  rcall kirim_asci
  ret
has3c2LED:
  ldi r20,'G'
  rcall kirim_asci
  ret
;client2: c LED
hasil2:
  ldi r20,0xc5
  rcall kirim_add
  lds r16,status3x
  cpi r16,0b00000000
  brne has2c2
  ldi r20,'L'
  rcall kirim_asci
  ret
has2c2:
  cpi r16,0b10000000
  brne has3c2
  ldi r20,'M'
  rcall kirim_asci
  ret
has3c2:
  ldi r20,'S'
  rcall kirim_asci
  ret
;;;;;;
;client2 set timing def
setTc:
  lds r16,status3x
  cpi r16,0b00000000
  brne shHasC1
  rcall TlargeC
  ret
shHasC1:
  cpi r16,0b10000000
  brne shHasC2
  rcall TmediumC
  ret
shHasC2:
  ;selebihnya dianggap small
  rcall TamallC
  ret

;;;;;;
;client3: c LED
hasil3LED:
  ldi r20,0xcd
  rcall kirim_add
  lds r16,traffic4
cpi r16,0b00000001
brne has2c3LED
ldi r20,'R'
rcall kirim_asci
ret

has2c3LED:
cpi r16,0b00000010
brne has3c3LED
ldi r20,'O'
rcall kirim_asci
ret

has3c3LED:
ldi r20,'G'
rcall kirim_asci
ret

;client3-c
hasilc3:
ldi r20,0xcc
rcall kirim_add
lds r16,status4x
cpi r16,0b00000000
brne has2c3
ldi r20,'L'
rcall kirim_asci
ret

has2c3:
cpi r16,0b10000000
brne has3c3
ldi r20,'M'
rcall kirim_asci
ret

has3c3:
ldi r20,'S'
rcall kirim_asci
ret

;>>client3 set timing def
setTd:
lds r16,status4x
cpi r16,0b00000000
brne sHasD1
rcall TlargeD
ret

sHasD1:
cpi r16,0b10000000
brne sHasD2
rcall TmediumD
ret

sHasD2:
:selebihnya dianggap small
rcall TsmallD
ret

;@view data timing GREEN
vGREEN:
ldi r20,0x8c
rcall kirim_add
lds r18,GREENs
rcall digi2
:.
ldi r20,0xc6
rcall kirim_add
lds r18,GREENm
rcall digi2
:.
ldi r20,0xcc
rcall kirim_add
lds r18,GREENl
rcall digi2
B - 21
; srmenu2:
rcall tmenu2
rcall vRED ; tampilkan setting waktu RED
ret

; set GREEN
rmenu3:
cpi r16,3
brne rmenuQ
lds r16,flag2
cpi r16,0xff
brne srmenu3
ldi r16,0
sts flag2,r16
rcall tmenu3
srmenu3:
rcall vGREEN
ret
; exit
rmenuQ:
ret

; @ grup tombol
grupT:
rcall tmenu
rcall tUPs
rcall tUPm
rcall tUPI
ret

; @ tombol menu
tMenu:
in r16,pinC
sbrc r16,0
rjmp not1

lds r16,flag3
cpi r16,0xff
breq bypas1
ldi r16,0xff ; lock flag3
sts flag3,r16 ; = act
ldi r16,0xff
sts flag2,r16 ; refresh txt menu

lds r16,cmenu
cpi r16,3 ; 3menu max
brne upMenu
ldi r16,1
sts cmenu,r16
ret

UpMenu:
inc r16
sts cmenu,r16
bypas1:
ret
not1:
ldi r16,0
sts flag3,r16
ret

; @ tombol UP small
tUPs:
in r16,pinD
sbrc r16,7
rjmp not2

lds r16,flag4
cpi r16,0xff
breq bypass2
ldi r16,0xff    ;lock flag3
sts flag4,r16 :=act
ldi r16,0xff    ;refresh txt menu
sts flag2,r16    :
ldi r16,0xff    ;cek posisi menu
ldi r16,cmenu   
cpi r16,2
brne cUP2
rjmp UPsmallR

cUP2:          
cpi r16,3
brne bypass2

:UPsmallG:    ;green
ldi r16,GREENs
ldi r16,30
brne upGREENs
ldi r16,1
sts GREENs,r16
ret
UpGREENs:      
inc r16
sts GREENs,r16
ret

:UPsmallR:    ;red
ldi r16,REDs
ldi r16,1
sts REDs,r16
ret
UpREDs:        
inc r16
sts REDs,r16

bypass2:       
ret

:not2:         
ldi r16,0
sts flag4,r16
ret

;@tombol UP medium

:UPm:          
in r16,pinD
sbrc r16,6
rjmp not3
:
ldi r16,flag5
ldi r16,0xff
breq bypass3
ldi r16,0xff    ;lock flag3
sts flag5,r16 :=act
ldi r16,0xff    ;refresh txt menu
sts flag2,r16    :
ldi r16,cmenu   
cpi r16,2
brne cUP3
rjmp UPmedR

cUP3:          
cpi r16,3
brne bypass3

:UPmedG:      ;green
ldi r16,GREENm
ldi r16,30
brne upGREENm
ldi r16,1  
sts GREENm,r16  
ret  
UpGREENm:  
  inc r16  
  sts GREENm,r16  
  ret  
UPmedR:  
  lds r16,REDm  
  cpi r16,30  
  brne upREDm  
  ldi r16,1  
  sts REDm,r16  
  ret  
UpREDm:  
  inc r16  
  sts REDm,r16  
  ret  
  ldi r16,0  
  sts flag5,r16  
  ret  
@tombol UP large  
  tUPI:  
    in r16,pinD  
    sbrc r16,5  
    rjmp not4  
    ;  
    lds r16,flag6  
    cpi r16,0xff  
    breq bypas4  
    ldi r16,0xff  
    ;lock flag3  
    sts flag6,r16  
    :=act  
    ldi r16,0xff  
    sts flag2,r16  
    ;refresh txt menu  
    ;cek posisi menu  
    lds r16,cmenu  
    cpi r16,2  
    brne cUP4  
    rjmp UPldgR  
  cUP4:  
    cpi r16,3  
    brne bypas4  
    ;  
  UPldgG:  
    ;green  
    lds r16,GREENI  
    cpi r16,30  
    brne upGREENI  
    ldi r16,1  
    sts GREENI,r16  
    ret  
UpGREENI:  
  inc r16  
  sts GREENI,r16  
  ret  
UPldgR:  
  ;red  
  lds r16,REDI  
  cpi r16,30  
  brne upREDI  
  ldi r16,1  
  sts REDI,r16  
  ret  
UpREDI:  
  inc r16  
  sts REDI,r16  
  ret  
  ldi r16,0  
  sts flag5,r16  
  ret
not4:
  ldi r16,0
  sts flag6,r16
  ret

@wait 74574
wait574:
  nop
  nop
  nop
  nop
  nop
  nop
  nop
  nop
  nop
  nop
  ret

<<<triger 574-1
TRIG574:
  rcall wait574
  out portA,r17
  rcall wait574
  cbi portb,0
  rcall wait574
  sbi portb,0
  rcall wait574
  ret

<<<menu LCD
tmenu0:
  ldi r20,0x80 ;baris1
  rcall kirim_add
  ldi zl,low(2*txt0)
  ldi zh,high(2*txt0)
  rcall kirim_txt
  ldi r20,0xc0 ;baris2
  rcall kirim_add
  ldi zl,low(2*txt1)
  ldi zh,high(2*txt1)
  rcall kirim_txt
  ret
tmenu1:
  ldi r20,0x80 ;baris1
  rcall kirim_add
  ldi zl,low(2*txt2)
  ldi zh,high(2*txt2)
  rcall kirim_txt
  ldi r20,0xc0 ;baris2
  rcall kirim_add
  ldi zl,low(2*txt2a)
  ldi zh,high(2*txt2a)
  rcall kirim_txt
  ret
tmenu2:
  ldi r20,0x80 ;baris1
  rcall kirim_add
  ldi zl,low(2*txt3)
  ldi zh,high(2*txt3)
  rcall kirim_txt
  ldi r20,0xc0 ;baris2
  rcall kirim_add
  ldi zl,low(2*txt3a)
ldi zh,high(2*tx3a)
rcall kirim_txt
ret

tmenu3:
ldi r20,0x80 ;baris1
rcall kirim_add
ldi zl,low(2*txt4)
ldi zh,high(2*txt4)
rcall kirim_txt
:
ldi r20,0xc0 ;baris2
rcall kirim_add
ldi zl,low(2*txt4a)
ldi zh,high(2*txt4a)
rcall kirim_txt
ret

;@tampilan 2digit+hex
digi2:
    mov r16,r18
    andi r16,0xf0
    swap r16
    ori r16,0x30
    rcall BCDhex
    :
    mov r16,r18
    andi r16,0x0f
    ori r16,0x30
    rcall BCDhex
    ret

://
BCDhex:
cpi r16,0x3a
brlo normalBCD
ldi r17,0x7
add r16,r17
normalBCD:
mov r20,r16
rcall kirim_asci
ret

;<<<test with timer
test1:
    lds r17,DTmp1
    : out portA,r16
    rcall trig574
    ret

;<<<
test0:
    ldi r17,0b00000000
    : out portA,r16
    rcall trig574
    ret

;<<<
testIO:
in r16,pinC
    bst r16,6 ;1
    bld r17,0
    : out portA,r17
    rcall trig574
    :
in r16,pinC
    bst r16,7 ;2
    bld r17,1
; out portA,r17
rcall trig574
ret

;;; initTimer1 >>>>
initTimer1:
ldi r16,0b00000100 ;enable int trm1 ovrflw
out TiMSK,r16
ldi r16.high(tmr1_value)
out TcnT1H,r16
ldi r16.low(tmr1_value)
out TcnT1L,r16
ldi r16.0b00000101
out TcCR1b,r16
sei
ret

;;; LCD stuff >>>>
;initLCD
initLCD:
ldi r20,0b00111000 ;function
rcall kirim_add
ldi r20,0b00001100 ;display
rcall kirim_add
ldi r20,0b00000110 ;entry
rcall kirim_add
ret

kirim_txt:
lpm
mov r17.r0
cpi r17.0x00
brne tulisTXT
ret
tulisTXT:
mov r20,r17
rcall kirim_asci
adiw Zh:Zi,1 ;Zi,1
rjmp kirim_txt

Kirim_add:
cbi portB,1 ;clr RSlcd/kirim add or command
sbi portB,2 ;setb ENLCD
rjmp tulis

; kirim_asci:
sbi portB,1 ;setb RSlcd/kirim ascii
sbi portB,2 ;setb ENLCD
tulis:
rcall delay40u 
out portA,r20 ;port LCD
rcall delay40u
cbi portB,2 ;clr ENlcd
rcall delay40u
ret

delay40u: ;delay kirim data ke LCD minimal 40uS
ldi r19,200
lop1:
dec r19
cpi r19,0
brne lop1
ret

delay1000m:
ldi r17,10 ;25
lop4:
ldi r18,200
lop3:
  ldi r19,200
lop2:
  dec r19
  cpi r19,0
  brne lop2
  dec r18
  cpi r18,0
  brne lop3
  dec r17
  cpi r17,0
  brne lop4
ret

; delay1m:
ldi r18,5
lop6:
  ldi r19,200
lop5:
  dec r19
  cpi r19,0
  brne lop5
  dec r18
  cpi r18,0
  brne lop6
ret

;>>> multiply timer utk delay LIMIT
lipatT1:
  lds r16,kali
  ldi r17,2
  cp r16,r17
  brne naik
  clr r16
  sts kali,r16
  lds r16,flagG1
  cpi r16,0xff
  brne LPT1q
  lds r16,stepRa
  cpi r16,0
    ;giliran hijau???
  brne LPT1q
    ;go ke orange???
  lds r16,bufPC
  sts status1,r16
LPT1q:
  ret
naik:
  inc r16
  sts kali,r16
  ret

;>>> countup index1,range0-2
randDT:
  lds r16,bilR1
    ;randomCD
  cpi r16,0x02
    ;range 0-2
  brge greatSame
  lds r16,bilR1
    ;randomCD
  inc r16
  sts bilR1,r16
    ;randomCD,r16
  ret
greatSame:
  ldi r16,0x00
    ;paksa=0
  sts bilR1,r16
    ;randomCD,r16
  ret

;>>> countup index2,test,1-4
randDT2:
lds r16, bilR2 ;randomCD
cpi r16, 0x04 ;range 1-4
brge greatSame2
lds r16, bilR2 ;randomCD
inc r16
sts bilR2, r16 ;randomCD, r16
ret
greatSame2:
  ldi r16, 0x01 ;paksa=1
  sts bilR2, r16 ;randomCD, r16
ret

;add index1+index2, random result range: 1-6
ranDRx:
  lds r17, bilR2
  lds r16, bilR1
  add r16, r17
  sts randomCD, r16
  ret

;=======================================
;interrupt vektor rutine
;=======================================
Ri_TMR1:
  push r19
  push r18
  push r17
  push r16
  rcall lipatT1
  rcall randDT
  rcall randDT2
  rcall ranDRx

  lds r16, DTtmp1 ;test LED with timing
  inc r16
  sts DTtmp1, r16
  :
  lds r16, flag1 ;togle flag1
  com r16
  sts flag1, r16
  :
  ldi r16, 0xff ;time for traffic giliran
  sts flag7, r16
  sts flag8, r16

;---
exitq1:
  rcall initTimer1
  ldi r16, 0bb00000100 ;tmr1 int flag diberi logika '1' biar kembali no(TOV1)
  out TiFR, r16
exit1:
  pop r16
  pop r17
  pop r18
  pop r19
  reti

;=======================================
;interrupt serial
;=======================================
;>>>konfigurasi usart
initUSART:
  ldi tmp, high(ubbr_value) ;set baud rate
  out ubrrh, tmp
  ldi tmp, low(ubbr_value)
out ubrrl,tmp
ldi tmp,(1<<txen)|(1<<rxen)|(1<<rxcie) ; enable receiver & transmitter
out ucarb,tmp
ldi tmp,(1<<ursel)|(3<<ucsz0) ; set frame format: 8bit data, 1 stop bit
out ucsrb,tmp
ret

>>> usart transmit data
usart_tx:
sbis ucsra,udre ; wait for empty transmit buffer
rjmp usart_tx
out udr,txbyte ; put data into buffer, sends the data
ret

>>> usart receive data
usart_rx:
sbis ucsra,rxc ; wait for data to be received
rjmp usart_rx
in rxbyte,udr ; get and return received data from buffer
ret

>>> USART routine
usart_rxc:
push r16
push r17
push r18
push r19
rcall usart_rx
rcall cekRX
pop r19
pop r18
pop r17
pop r16
reti

>>> enable int serial
settingEN:
ldi r21,0b10000000 ; enable flag receive
out ucsra,r21
sei
ret

settingDIS:
ldi r21,0b00000000 ; disable flag receive
out ucsra,r21
sei
ret

>>> cek header
cekRX:
lds r16,flagR5d
cpi r16,0xff
breq bsvCode4
; /
lds r16,flagR5c
cpi r16,0xff
breq bsvCode3
; /
lds r16,flagR5b
cpi r16,0xff
breq bsvCode2
; /
lds r16,flagR3d ; header client3(b4h) terdeteksi
cpi r16,0xff
breq bloadDTd
; /
lds r16,flagR3c ; header client2(b3h) terdeteksi
cpi r16,0xff

breq loadDTc
; header client1(b2h) terdeteksi
lds r16,flagR3b
cli r16,0xff
breq loadDTb
; header d4h terdeteksi
lds r16,flagR2
cli r16,0xff
breq cekH3b
; header 2dh terdeteksi
lds r16,flagR1
cli r16,0xff
breq cekH2

cekH1:
  mov r16,rxbyte
  cli r16,0x8a
  brne breset
  ldi r16,0xff
  sts flagR1,r16
  rjmp done

>>> buffer jump
bloadDTd:
  rjmp loadDTd
bsvCode2:
  rjmp svCode2
bsvCode3:
  rjmp svCode3
bsvCode4:
  rjmp svCode4
breset:
  rjmp reset

>>> buffer jump
cekH2:
  mov r16,rxbyte
  cli r16,0x9b
  brne breset
  ldi r16,0xff
  sts flagR2,r16
  rjmp done

cekH3b:
  mov r16,rxbyte
  sts dtUSER,r16
  cli r16,0xd7
  brne cekH3c
  ldi r16,0xff
  sts flagR3b,r16
  rjmp done

cekH3c:
  cli r16,0xd8
  brne cekH3d
  ldi r16,0xff
  sts flagR3c,r16
  rjmp done

cekH3d:
  cli r16,0xd9
  brne reset
  ldi r16,0xff
  sts flagR3d,r16
  rjmp done

loadDTb:
  mov r16,rxbyte
  sts status2,r16
  ldi r16,0xff
  sts flagR5b,r16
  rjmp done

loadDTc:
  mov r16,rxbyte
  sts status3,r16
ldi r16,0xff
sts flagR5c,r16
rjmp done

loadDTd:
mov r16,rxbyte
sts status4,r16
ldi r16,0xff
sts flagR5d,r16
rjmp done
dsCode2:
mov r16,rxbyte
sts code2,r16
ldi r16,0xff
sts flagC2,r16

;encrypt paket completed
rjmp reset
dsCode3:
mov r16,rxbyte
sts code3,r16
ldi r16,0xff
sts flagC3,r16

;encrypt paket completed
rjmp reset
dsCode4:
mov r16,rxbyte
sts code4,r16
ldi r16,0xff
sts flagC4,r16

;encrypt paket completed
rjmp reset
reset:
ldi r16,0
sts flagR1,r16
sts flagR2,r16
sts flagR3b,r16
sts flagR3c,r16
sts flagR3d,r16
sts flagR4,r16
sts flagR5b,r16
sts flagR5c,r16
sts flagR5d,r16
done:
ret
txt0:
.db " Rendy Traffic ",0
txt1:
.db " Miniatur ",0
txt2:
.db "M(-x) Ca(-x) ",0 ;dalam kurung berisi status[sepi/S,sedang/M,padat/L,red/R,org/O,green/G]
txt2a:
.db " Cb(-x) Cc(-x) ",0
txt3:
.db "RED: S(-) ",0
txt3a:
.db " M(-) L(-) ",0
txt4:
.db "GREEN: S(-) ",0
txt4a:
.db " M(-) L(-) ",0

;table random encrypted,test
cryptCode:
.db 0x07,0x03,0x09,0x73,0x39,0x79
LAMPIRAN C

Listing Program Slave
; traffic client source code

.include "J:\AVR\m8535def.inc"
def tmp=r24
def txbyte=r25
def rxbyte=r18

equ fclock=11059200
equ baud_rate=1200 ; 2400
equ ubbr_value=(fclock/(16*baud_rate))-1

equ tmr1_value=0xd5d0 ; 0xdae8 ; eae8 ; set waktu tmr1
equ flag1=$60
equ DTtmp1=$61
equ dataTX=$62
.equ flagR1=$63
equ flagR2=$64
equ flagR3=$65
equ flag2=$66
equ flagLS=$67
equ flagR4=$68
equ flagD1=$69
equ kai=$6a
equ flagD2=$6b
equ bufPC=$6c
.equ bufPA=$6d
.equ flagL1=$6e
.equ randomCD=$6f ; random code: 1-5
.equ dataTXx=$70 ; test
equ TranCD=$71
equ TranCD=$71
equ flagR5=$72
equ codeEC=$73
equ bufPAx=$74
.equ flagEC=$75
.equ bilR1=$76
.equ bilR2=$77

.org 0x0000
rjmp init

.org 0x0008 ; vector int tmr1 overflow
rjmp Ri_TMR1

.org 0x000b
rjmp usart_rxc

init:
ldi r16, low(ramend)
out spl, r16
ldi r16, high(ramend)
out sph, r16

;---

;--- setting fungsi port
ldi r16, 0xff
ldi r19, 0x00
out ddrA, r16 ; set port A sebagai output
out portA, r16
out ddrB, r16 ; set port B sebagai output
out portB, r16
out ddrC, r19 ; set port C sebagai masukan pull up
out portC, r16
out ddrD, r16 ; set port D sebagai output
out portD, r16

ldi r16, 0x0
sts bilR1, r16
sts flagEC, r16
ldi r16, 0x1
sts bilR2, r16

;--end setting port

ldi r16, 0
sts bufPAx, r16
sts flagR5, r16
sts flagR1, r16
sts flagR2, r16
sts flagR3, r16
sts flagR4, r16
sts flag2, r16
sts flagLS, r16
sts flagD1, r16
sts flagD2, r16
sts bufPC, r16
sts bufPA, r16
sts flagL1, r16

;--init register
rcall initTimer1
rcall initUSART
rcall settingEN
sts DTtmp1, r19
sbi portD.2 ; disable TX
cbi portD.2 ; enable TX

;===================================

UTAMA:
;
rcall testIO
;
rcall test1
rcall loadLMT
rcall kirimStatus
rcall delay1m
;
rcall outkePA
rjmp UTAMA

;=================================================================

:decript data lampu dari master
decrpDmstq:
ret
decrpDmst:
lds r16, flagEC
cpi r16, 0xff
brne decrpDmstq
clr r16
sts flagEC, r16
;
lds r18, codeEC
rcall setTabel
lds r16, bufPA
rcall run_dcript
sts bufPAx, r16
ret

:encript data
cncrDT:
lds r18, randomCD
sts TranCD, r18
rcall setTabel
rcall run_ccript
ret

;load encript code,test
setTabel:
ldi zl, low(2*criptCode)
ldi zh, high(2*criptCode)
;load encript table,test
run_cript:
    ;tentukan baris tabel? yg akan diakses
dec r18
    cpi r18,0
    breq goCrip
    adiw Zh:Zl,1 ;cek nilai register "randomCD"
rjmp run_cript
goCrip:
lpm
    mov r17,r0
    lds r16,dataTX
    eor r16,r17
    sts dataTXx,r16
    ret

;load decript table,test
run_dcript:
dec r18
    cpi r18,0
    breq godCrip
    adiw Zh:Zl,1 ;cek nilai register "randomCD"
rjmp run_dcript
godCrip:
lpm ;
    mov r17,r0
    eor r16,r17
    ret

proteksi1:
    mov r17,r16
    andi r17,0b000000000001
    cpi r17,0b000000000001
    bne proteksi2
    ldi r16,0b000000000001
    rjmp gabungX
    ///
proteksi2:
    mov r17,r16
    andi r17,0b000000000011
    cpi r17,0b000000000010
    bne proteksi3
    ldi r16,0b000000000010
    rjmp gabungX
    ///
proteksi3:
    mov r17,r16
    andi r17,0b0000000000111
    cpi r17,0b0000000000110
    bne gabungx
    ldi r16,0b0000000000100
    rjmp gabungX
    ///

;@kirim ke LED
outkePA:
    rcall decrpDmst
    lds r16,bufPAx
    andi r16,0b00000000000001
    rjmp proteksi1

GabungX:
    lds r17,dataTX ;data limit
    andi r17,0b10000000
    or r16,r17
out portA,r16
ret

;kirim ke client
kirimDum:
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  ldi txbyte,0xaa
  rcall usart_tx
  nop
  nop
  nop
  nop
ret

;@load limit status
loadLMTq:
  ret
loadLMT:
  lds r16,flagL1
  cpi r16,0xff
  brne loadLMTq
  clr r16
  sts flagL1,r16
  in r16,pinC
  andi r16,0b11000000
  sts bufPC,r16
  cpi r16,0b00000000
  ;limit1-2 ketekan
  breq cekv2
  lds r16,flagD1
  cpi r16,0xff
  breq cekNOq
  clr r16
  sts kali,r16
  sts flagD2,r16
  ser r16
  sts flagD1,r16
  ;lock flagD1
  ldi r16,0b11000011
  sts dataTX,r16
  ret

cekv2:
  cpi r16,0b10000000
  ;hanya limit1 ketekan
  brne cekv3
  lds r16,flagD2
  cpi r16,0xff
  breq cekNOq
  clr r16
  sts kali,r16
  sts flagD1,r16
  ser r16
  sts flagD2,r16
  ;lock flagD2
  ldi r16,0b11000001
  sts dataTX,r16
  ret

cekv3:
  cpi r16,0b01000000
  ;hanya limit2 ketekan=abaikan
  brne cekNO
  ldi r16,0b11000000
  sts bufPC,r16
clr r16
sts flagD1,r16
sts flagD2,r16
ret
cekNO:
ori r16,0b00000011
sts dataTX,r16
clr r16
sts flagD1,r16
sts flagD2,r16
cekNOq:
ret

;kirim paket
kirimStatusQ:
sbi portD,2 ;disable TX
ret
kirimStatus:
lds r16,flag2
cpi r16,0xff
brne kirimStatusQ
ldi r16,0
sts flag2,r16
/*
rcall encrDT
rcall setingDIS
.:
cbi portD,2 ;enable TX
rcall payload
rcall payload
rcall payload
rcall payload
.:
rcall payGilir
rcall payGilir
.:
sbi portD,2 ;disable TX
.:
rcall setingEN
ret
payload:
.:
ldi txbyte,0x8a ;r16
rcall usart_tx
nop
nop
nop
.:
ldi txbyte,0x9b ;r16
rcall usart_tx
nop
nop
nop
.<<ganti client=ganti setting>>;1
ldi txbyte,0xd9
rcall usart_tx
nop
nop
nop
.:
lds txbyte,dataTXx ;test
rcall usart_tx
nop

C - 5
nop
nop
\/
lds txbyte,TranCD ;code random, test
rcall usart_tx
nop
nop
nop
ret

payGilir:
\/
ldi txbyte,0x8a ;r16
rcall usart_tx
nop
nop
nop
\/
ldi txbyte,0x9b ;r16 ;d4=kode transmit dr client, e5=kode transmit dr master
rcall usart_tx
nop
nop
nop
\/<<<ganti client=ganti setting>>>/2
ldi txbyte,0xe9
rcall usart_tx
nop
nop
nop
\/
ldi txbyte,0xaa ;r16
rcall usart_tx
nop
nop
nop
ret

;<<<<test with timer

\<<<
test1:
lds r16,DTmp1
out portA,r16
ret
\<<<
test0:
ldi r16,0b01010100
out portA,r16
ret
\<<<
testIO:
in r16,pinC
bst r16,6
bld r17,0
out portA,r17
\/
in r16,pinC
bst r16,7
bld r17,1
out portA,r17
ret
\<<< initTimer1 >>>>

\<<< initTimer1 >>>>
initTimer1:
ldi r16,0b00000100 ;enable int trm1 ovflw
out TIMSK,r16
ldi r16,high(tmr1_value)
out TcnT1H,r16
ldi r16,low(tmr1_value)
out TcnT1L,r16
ldi r16,0b000000101
out TcCR1b,r16
sei
ret

; delay40u:
  ldi r19,200
  lop1:
    dec r19
    cpi r19,0
    brne lop1
  ret

; delay1000m:
  ldi r17,2
  lop4:
    ldi r18,200
    lop3:
      ldi r19,200
      lop2:
        dec r19
        cpi r19,0
        brne lop2
        dec r18
        cpi r18,0
        brne lop3
        dec r17
        cpi r17,0
        brne lop4
      ret

; delay1m:
  ldi r18,5
  lop6:
    ldi r19,200
  lop5:
    dec r19
    cpi r19,0
    brne lop5
    dec r18
    cpi r18,0
    brne lop6
  ret

;>>>multiply timer utk delay LIMIT
lipatT1:
  lds r16,kali
  ldi r17,2
  cp r16,r17
  brne naik
  clr r16
  sts kali,r16
  lds r16,bufPC
  ori r16,0b00000011
  sts dataTX,r16
  ret

naik:
  inc r16
  sts kali,r16
  ret
C - 8
;==============================================
;interrupt serial
;==============================================

;>>>konfigurasi usart
initUSART:
        ldi tmp,high(ubbr_value) ; set baud rate
        out ubrrh,tmp
        ldi tmp,low(ubbr_value)
        out ubrrl,tmp
        ldi tmp,0b10011000
        ; (1<<rxen)|(1<<txen)|(1<<rxcie)
        out ucsrb,tmp
        ldi tmp,0b10000110
        ; (1<<ursel)|(3<<ucsz0)
        out ucsrc,tmp
        ret

;>>>usart transmit data
usart_tx:
        sbis ucsra,udre
        rjmp usart_tx
        out udr,txbyte
        ; put data into buffer, sends the data
        ret

;>>>usart receive data
usart_rx:
        sbis ucsra,rxc
        rjmp usart_rx
        in rxbyte,udr
        ; get and return received data from buffer
        ret

;>>>USART rutine
usart_rxc:
        push r16
        push r17
        push r18
        push r19
        rcall usart_rx
        rcall cekRX
        pop r19
        pop r18
        pop r17
        pop r16
        reti

;>>>enable int serial
setingEN:
        ldi r21,0b10000000
        out ucsra,r21
        ; enable flag receive k
        sei
        ret

setingDIS:
        ldi r21,0b00000000
        out ucsra,r21
        ; disable flag receive k
        sei
        ret

;>>>cek header
cekRX:
        lds r16,flagR5
        cpi r16,0xff
        breq getKey
        ;
        lds r16,flagR4
        cpi r16,0xff
        breq loadEND
        ;
        lds r16,flagR3
        cpi r16,0xff

C - 9
breq loadDT
lds r16,flagR2
cpi r16,0xff
breq cekH3
lds r16,flagR1
cpi r16,0xff
breq cekH2
.cekH1:
mov r16(rxbyte

cpi r16,0x8a

brne reset
ldi r16,0xff
sts flagR1,r16
rjmp done
.cekH2:
mov r16(rxbyte

cpi r16,0x9b

brne reset
ldi r16,0xff
sts flagR2,r16
rjmp done
.cekH3:
mov r16(rxbyte

;c<<ganti client=ganti setting>>>3

cpi r16,0xdd

brne ganti

ldi r16,0xff
sts flagR3,r16
rjmp done
.ganti:
;c<<ganti client=ganti setting>>>4

cpi r16,0xe8

brne reset
ldi r16,0xff
sts flagR4,r16
rjmp done
.loadDT:
mov r16(rxbyte

sts bufPA,r16
ldi r16,0xff
sts flagR5,r16
rjmp done
.getKey:
mov r16(rxbyte

sts codeEC,r16
ldi r16,0xff
sts flagEC,r16
rjmp reset
.loadEND:
ldi r16,0xff
sts flag2,r16
;kirim balasan giliran
.reset:
ldi r16,0
sts flagR1,r16
sts flagR2,r16
sts flagR3,r16
sts flagR4,r16
sts flagR5,r16
.done:
ret

.tabel random encrpted,test
criptCode:
.db 0x07,0x03,0x09,0x73,0x39,0x79
LAMPIRAN D

Data Sheet Komponen

TLP & RLP 315  D - 1
ULN2803       D - 2
LATCH 74LS574 D - 8
ATMega 8535   D - 13
Octal High Voltage, High Current Darlington Transistor Arrays

The eight NPN Darlington connected transistors in this family of arrays are ideally suited for interfacing between low logic level digital circuitry (such as TTL, CMOS or PMOS/NMOS) and the higher current/voltage requirements of lamps, relays, printer hammers or other similar loads for a broad range of computer, industrial, and consumer applications. All devices feature open-collector outputs and free-wheeling clamp diodes for transient suppression.

The ULN2803 is designed to be compatible with standard TTL families while the ULN2804 is optimized for 6 to 15 volt high level CMOS or PMOS.

MAXIMUM RATINGS ($T_A = 25^\circ\text{C}$ and rating apply to any one device in the package, unless otherwise noted.)

<table>
<thead>
<tr>
<th>Rating</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Voltage</td>
<td>$V_O$</td>
<td>50</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage (Except ULN2801)</td>
<td>$V_I$</td>
<td>30</td>
<td>V</td>
</tr>
<tr>
<td>Collector Current – Continuous</td>
<td>$I_C$</td>
<td>500</td>
<td>mA</td>
</tr>
<tr>
<td>Base Current – Continuous</td>
<td>$I_B$</td>
<td>25</td>
<td>mA</td>
</tr>
<tr>
<td>Operating Ambient Temperature Range</td>
<td>$T_A$</td>
<td>0 to +70</td>
<td>$^\circ\text{C}$</td>
</tr>
<tr>
<td>Storage Temperature Range</td>
<td>$T_{stg}$</td>
<td>−55 to +150</td>
<td>$^\circ\text{C}$</td>
</tr>
<tr>
<td>Junction Temperature</td>
<td>$T_J$</td>
<td>125</td>
<td>$^\circ\text{C}$</td>
</tr>
</tbody>
</table>

$R_{JA} = 55\ \text{C/W}$
Do not exceed maximum current limit per driver.

ORDERING INFORMATION

<table>
<thead>
<tr>
<th>Device</th>
<th>Input Compatibility</th>
<th>$V_{CE\text{(Max)}}$</th>
<th>$I_C\text{(Max)}$</th>
<th>Operating Temperature Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULN2803A</td>
<td>TTL, 5.0 V CMOS</td>
<td>50 V</td>
<td>500 mA</td>
<td>$T_A = 0$ to $+70^\circ\text{C}$</td>
</tr>
<tr>
<td>ULN2804A</td>
<td>6 to 15 V CMOS, PMOS</td>
<td>50 V</td>
<td>500 mA</td>
<td>$T_A = 0$ to $+70^\circ\text{C}$</td>
</tr>
</tbody>
</table>
ULN2803 ULN2804

**ELECTRICAL CHARACTERISTICS**  \( (T_A = 25^\circ C, \text{unless otherwise noted}) \)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Symbol</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output Leakage Current (Figure 1)</td>
<td>( I_{CEX} )</td>
<td>–</td>
<td>–</td>
<td>100</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>( V_C = 50 , V, , T_A = +70^\circ C )</td>
<td>All Types</td>
<td>–</td>
<td>–</td>
<td>50</td>
<td>–</td>
</tr>
<tr>
<td>( V_C = 50 , V, , T_A = +125^\circ C )</td>
<td>All Types</td>
<td>–</td>
<td>–</td>
<td>50</td>
<td>–</td>
</tr>
<tr>
<td>( V_C = 50 , V, , T_A = +70^\circ C, , V_I = 6.0 , V )</td>
<td>ULN2802</td>
<td>–</td>
<td>–</td>
<td>500</td>
<td>–</td>
</tr>
<tr>
<td>( V_C = 50 , V, , T_A = +70^\circ C, , V_I = 1.0 , V )</td>
<td>ULN2804</td>
<td>–</td>
<td>–</td>
<td>500</td>
<td>–</td>
</tr>
<tr>
<td>Collector–Emitter Saturation Voltage (Figure 2)</td>
<td>( V_{CE(sat)} )</td>
<td>–</td>
<td>1.1</td>
<td>1.6</td>
<td>( V )</td>
</tr>
<tr>
<td>( I_C = 350 , mA, , I_E = 500 , \mu A )</td>
<td>All Types</td>
<td>–</td>
<td>0.95</td>
<td>1.3</td>
<td>–</td>
</tr>
<tr>
<td>( I_C = 200 , mA, , I_E = 350 , \mu A )</td>
<td>All Types</td>
<td>–</td>
<td>0.85</td>
<td>1.1</td>
<td>–</td>
</tr>
<tr>
<td>( I_C = 100 , mA, , I_E = 250 , \mu A )</td>
<td>All Types</td>
<td>–</td>
<td>0.85</td>
<td>1.1</td>
<td>–</td>
</tr>
<tr>
<td>Input Current – On Condition (Figure 4)</td>
<td>( I_{(on)} )</td>
<td>–</td>
<td>0.82</td>
<td>1.25</td>
<td>mA</td>
</tr>
<tr>
<td>( V_I = 17 , V )</td>
<td>ULN2802</td>
<td>–</td>
<td>0.53</td>
<td>1.35</td>
<td>–</td>
</tr>
<tr>
<td>( V_I = 3.85 , V )</td>
<td>ULN2803</td>
<td>–</td>
<td>0.35</td>
<td>0.5</td>
<td>–</td>
</tr>
<tr>
<td>( V_I = 5.0 , V )</td>
<td>ULN2804</td>
<td>–</td>
<td>0.35</td>
<td>0.5</td>
<td>–</td>
</tr>
<tr>
<td>( V_I = 12 , V )</td>
<td>ULN2804</td>
<td>–</td>
<td>1.0</td>
<td>1.45</td>
<td>–</td>
</tr>
<tr>
<td>Input Voltage – On Condition (Figure 5)</td>
<td>( V_{(on)} )</td>
<td>–</td>
<td>–</td>
<td>13</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 300 , mA )</td>
<td>ULN2802</td>
<td>–</td>
<td>–</td>
<td>3.0</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 200 , mA )</td>
<td>ULN2803</td>
<td>–</td>
<td>–</td>
<td>2.4</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 125 , mA )</td>
<td>ULN2804</td>
<td>–</td>
<td>–</td>
<td>2.7</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 125 , mA )</td>
<td>ULN2804</td>
<td>–</td>
<td>–</td>
<td>3.0</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 275 , mA )</td>
<td>ULN2804</td>
<td>–</td>
<td>–</td>
<td>7.0</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 350 , mA )</td>
<td>ULN2804</td>
<td>–</td>
<td>–</td>
<td>8.0</td>
<td>–</td>
</tr>
<tr>
<td>Input Current – Off Condition (Figure 3)</td>
<td>( I_{(off)} )</td>
<td>–</td>
<td>50</td>
<td>100</td>
<td>–</td>
</tr>
<tr>
<td>( I_C = 500 , \mu A, , T_A = +70^\circ C )</td>
<td>All Types</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>DC Current Gain (Figure 2)</td>
<td>( h_{FE} )</td>
<td>1000</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>( V_{CE} = 2.0 , V, , I_C = 350 , mA )</td>
<td>ULN2801</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Input Capacitance</td>
<td>( C_I )</td>
<td>–</td>
<td>15</td>
<td>26</td>
<td>( \text{pF} )</td>
</tr>
<tr>
<td>Turn-On Delay Time</td>
<td>( t_{on} )</td>
<td>–</td>
<td>0.25</td>
<td>1.0</td>
<td>( \mu s )</td>
</tr>
<tr>
<td>( (50% , E_I \text{ to } 50% , E_O) )</td>
<td>–</td>
<td>–</td>
<td>0.25</td>
<td>1.0</td>
<td>( \mu s )</td>
</tr>
<tr>
<td>Turn-Off Delay Time</td>
<td>( t_{off} )</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>( (50% , E_I \text{ to } 50% , E_O) )</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Clamp Diode Leakage Current (Figure 6)</td>
<td>( I_R )</td>
<td>–</td>
<td>–</td>
<td>50</td>
<td>( \mu A )</td>
</tr>
<tr>
<td>( T_A = +25^\circ C )</td>
<td>–</td>
<td>–</td>
<td>100</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>( T_A = +70^\circ C )</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>Clamp Diode Forward Voltage (Figure 7)</td>
<td>( V_F )</td>
<td>–</td>
<td>1.5</td>
<td>2.0</td>
<td>( V )</td>
</tr>
<tr>
<td>( I_R = 350 , mA )</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
</tbody>
</table>
ULN2803 ULN2804

TEST FIGURES

(See Figure Numbers in Electrical Characteristics Table)

Figure 1.

Figure 2.

Figure 3.

Figure 4.

Figure 5.

Figure 6.

Figure 7.
ULN2803 ULN2804

TYPICAL CHARACTERISTIC CURVES – $T_A = 25^\circ$C, unless otherwise noted

Output Characteristics

**Figure 8. Output Current versus Saturation Voltage**

**Figure 9. Output Current versus Input Current**

Input Characteristics

**Figure 10. ULN2803 Input Current versus Input Voltage**

**Figure 11. ULN2804 Input Current versus Input Voltage**

**Figure 12. Representative Schematic Diagrams**

![Schematic Diagrams](image)
DM74LS574
Octal D-Type Flip-Flop with 3-STATE Outputs

General Description
The DM74LS574 is a high speed low power octal flip-flop with a buffered common Clock (CP) and a buffered common Output Enable (OE). The information presented to the D inputs is stored in the flip-flops on the LOW-to-HIGH Clock (CP) transition.
This device is functionally identical to the DM74LS374 except for the pinouts.

Ordering Code:

<table>
<thead>
<tr>
<th>Order Number</th>
<th>Package Number</th>
<th>Package Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DM74LS574WM</td>
<td>M20B</td>
<td>20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide</td>
</tr>
<tr>
<td>DM74LS574N</td>
<td>N20A</td>
<td>20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide</td>
</tr>
</tbody>
</table>

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Logic Symbol

Connection Diagram

Truth Table

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dn</td>
<td>CP</td>
</tr>
<tr>
<td>H</td>
<td>-</td>
</tr>
<tr>
<td>L</td>
<td>-</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

H = HIGH Voltage Level
L = LOW Voltage Level
X = Immotential
Z = High Impedance
-= HIGH-to-LOW Clock (CP) transition
Functional Description
The DM74LS574 consists of eight edge-triggered flip-flops with individual D-type inputs and 3-STATE true outputs. The buffered clock and buffered Outputs Enable are common to all flip-flops. The eight flip-flops will store the state of their individual D inputs that meet the setup and hold times requirements on the LOW-to-HIGH Clock (CP) transition. With the Output Enable (OE) LOW, the contents of the eight flip-flops are available at the outputs. When the OE is HIGH, the outputs go to the high impedance state. Operation of the OE input does not affect the state of the flip-flops.

Logic Diagram
### Absolute Maximum Ratings (Note 1)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Min</th>
<th>Nom</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>7V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Voltage</td>
<td>7V</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Free Air Temperature Range</td>
<td>0°C to +70°C</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Storage Temperature Range</td>
<td>-65°C to +150°C</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

### Recommended Operating Conditions

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Nom</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>Supply Voltage</td>
<td>4.75</td>
<td>5</td>
<td>5.25</td>
<td>V</td>
</tr>
<tr>
<td>VIL</td>
<td>LOW Level Input Voltage</td>
<td>0.8</td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>IOH</td>
<td>HIGH Level Output Current</td>
<td>-2.6</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>IOL</td>
<td>LOW Level Output Current</td>
<td>24</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>TA</td>
<td>Free Air Operating Temperature</td>
<td>0</td>
<td>70</td>
<td></td>
<td>°C</td>
</tr>
<tr>
<td>tH (H)</td>
<td>Setup Time HIGH or LOW</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>tH (L)</td>
<td>DN to CP</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>tH (H)</td>
<td>Hold Time HIGH or LOW</td>
<td>0</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>tH (L)</td>
<td>DN to CP</td>
<td>0</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>tW (H)</td>
<td>CP Pulse Width</td>
<td>15</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>tW (L)</td>
<td>HIGH or LOW</td>
<td>15</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

### Electrical Characteristics

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ (Note 2)</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIL</td>
<td>Input Clamp Voltage</td>
<td>2.4</td>
<td>3.3</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>VIL</td>
<td>LOW Level Output Voltage</td>
<td>0.35</td>
<td>0.5</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>Ii</td>
<td>Input Current @ Max Input Voltage</td>
<td>0.25</td>
<td>0.4</td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>IH</td>
<td>HIGH Level Input Current</td>
<td>0.1</td>
<td></td>
<td>0.1</td>
<td>mA</td>
</tr>
<tr>
<td>IL</td>
<td>LOW Level Input Current</td>
<td>20</td>
<td></td>
<td>2.7V</td>
<td>µA</td>
</tr>
<tr>
<td>IOH2</td>
<td>OFF-State Output Current with</td>
<td>20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOH1</td>
<td>HIGH Level Output Voltage Applied</td>
<td>20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>20</td>
<td></td>
<td>2.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>20</td>
<td></td>
<td>2.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>LOW Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL1</td>
<td>OFF-State Output Current with</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
<tr>
<td>IOL2</td>
<td>HIGH Level Output Voltage Applied</td>
<td>-20</td>
<td></td>
<td>0.4V</td>
<td>µA</td>
</tr>
</tbody>
</table>
## Switching Characteristics

$V_{CC} = +5.0V$, $T_A = +25^\circ C$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>$R_L = 2,k\Omega$, $C_L = 45,pF$</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{\text{MAX}}$</td>
<td>Maximum Clock Frequency</td>
<td>35</td>
<td>MHz</td>
</tr>
<tr>
<td>$\tau_{\text{PHL}}$</td>
<td>Propagation Delay CP to On</td>
<td>28</td>
<td>ns</td>
</tr>
<tr>
<td>$\tau_{\text{PHL}}$</td>
<td>Output Enable Time</td>
<td>28</td>
<td>ns</td>
</tr>
<tr>
<td>$\tau_{\text{PHL}}$</td>
<td>Output Disable Time</td>
<td>20</td>
<td>ns</td>
</tr>
<tr>
<td>$\tau_{\text{PLZ}}$</td>
<td></td>
<td>25</td>
<td>ns</td>
</tr>
</tbody>
</table>
Physical Dimensions inches (millimeters) unless otherwise noted (Continued)

Fairchild does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and Fairchild reserves the right at any time without notice to change said circuitry and specifications.

LIFE SUPPORT POLICY

FAIRCHILD’S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.
Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 130 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
  - 8K Bytes of In-System Self-Programmable Flash
    Endurance: 10,000 Write/Erase Cycles
  - Optional Boot Code Section with Independent Lock Bits
  - In-System Programming by On-chip Boot Program
  - True Read-While-Write Operation
  - 512 Bytes EEPROM
    Endurance: 100,000 Write/Erase Cycles
  - 512 Bytes Internal SRAM
  - Programming Lock for Software Security
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    - 8 Single-ended Channels
    - 7 Differential Channels for TQFP Package Only
    - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x for TQFP Package Only
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega8535L
  - 4.5 - 5.5V for ATmega8535
- Speed Grades
  - 0 - 8 MHz for ATmega8535L
  - 0 - 16 MHz for ATmega8535

Summary

ATmega8535
ATmega8535L

Note: This is a summary document. A complete document is available on our Web site at www.atmel.com.
Pin Configurations

Figure 1. Pinout ATmega8535

![Pinout Diagram]

**Note:** MLF Bottom pad should be soldered to ground.

**Disclaimer**

Typical values contained in this data sheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.
Overview

The ATmega8535 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing instructions in a single clock cycle, the ATmega8535 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 2. Block Diagram
The AVR core combines a rich instruction set with 32 general purpose working registers. All 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega8535 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 512 bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain in TQFP package, a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the asynchronous timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega8535 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega8535 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, In-Circuit Emulators, and evaluation kits.

**AT90S8535 Compatibility**

The ATmega8535 provides all the features of the AT90S8535. In addition, several new features are added. The ATmega8535 is backward compatible with AT90S8535 in most cases. However, some incompatibilities between the two microcontrollers exist. To solve this problem, an AT90S8535 compatibility mode can be selected by programming the S8535C fuse. ATmega8535 is pin compatible with AT90S8535, and can replace the AT90S8535 on current Printed Circuit Boards. However, the location of fuse bits and the electrical characteristics differs between the two devices.

**AT90S8535 Compatibility Mode**

Programming the S8535C fuse will change the following functionality:

- The timed sequence for changing the Watchdog Time-out period is disabled. See “Timed Sequences for Changing the Configuration of the Watchdog Timer” on page 45 for details.
- The double buffering of the USART Receive Register is disabled. See “AVR USART vs. AVR UART – Compatibility” on page 146 for details.
Pin Descriptions

$V_{CC}$
Digital supply voltage.

GND
Ground.

Port A (PA7..PA0)
Port A serves as the analog inputs to the A/D Converter.
Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used.
Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability.
When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7..PB0)
Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability.
As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.
Port B also serves the functions of various special features of the ATmega8535 as listed on page 60.

Port C (PC7..PC0)
Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability.
As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.
Port C also serves the functions of various special features of the ATmega8535 as listed on page 64.

RESET
Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 37. Shorter pulses are not guaranteed to generate a reset.

XTAL1
Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

XTAL2
Output from the inverting Oscillator amplifier.

AVCC
AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to $V_{CC}$ even if the ADC is not used. If the ADC is used, it should be connected to $V_{CC}$ through a low-pass filter.

AREF
AREF is the analog reference pin for the A/D Converter.
Resources

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.
About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C Compiler documentation for more details.
## Register Summary

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xF8</td>
<td>SRW0</td>
<td>T</td>
<td>T</td>
<td></td>
<td>S</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>0x1E</td>
<td>SPPH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SP0</td>
<td>12</td>
</tr>
<tr>
<td>0x1D</td>
<td>SPPH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>SP0</td>
<td>12</td>
</tr>
<tr>
<td>0x0C</td>
<td>OCR0</td>
<td>Timer/Counter Output Compare Register</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>85</td>
</tr>
<tr>
<td>0x08</td>
<td>GICR</td>
<td>INT1</td>
<td>INT0</td>
<td>INT2</td>
<td></td>
<td>IC11</td>
<td>OC1A</td>
<td>OC1B</td>
<td>TOV1</td>
<td>TOV0</td>
</tr>
<tr>
<td>0x04</td>
<td>GICR</td>
<td>INT1</td>
<td>INT0</td>
<td>INT2</td>
<td></td>
<td>IC11</td>
<td>OC1A</td>
<td>OC1B</td>
<td>TOV1</td>
<td>TOV0</td>
</tr>
<tr>
<td>0x39</td>
<td>TIM5</td>
<td>OCF1</td>
<td>TOV1</td>
<td>TOV1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>113</td>
</tr>
<tr>
<td>0x37</td>
<td>TIM5</td>
<td>OCF1</td>
<td>TOV1</td>
<td>TOV1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>113</td>
</tr>
<tr>
<td>0x2D</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x2C</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x28</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x18</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x08</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x01</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0xF8</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
<tr>
<td>0x1E</td>
<td>OCR0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>218</td>
</tr>
</tbody>
</table>

*Note: OCR0 is an Output Compare Register.*
Register Summary (Continued)

<table>
<thead>
<tr>
<th>Address</th>
<th>Name</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>TWBR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>181</td>
</tr>
</tbody>
</table>

Notes:
1. Refer to the USART description for details on how to access UBRRH and UCSRC.
2. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
3. Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O Register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers 0x00 to 0x1F only.
## Instruction Set Summary

<table>
<thead>
<tr>
<th>Mnemonics</th>
<th>Operands</th>
<th>Description</th>
<th>Operation</th>
<th>Flags</th>
<th>#Clocks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ARITHMETIC AND LOGIC INSTRUCTIONS</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>Rd, Rr</td>
<td>Add two Registers</td>
<td>Rd ← Rd + Rr</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>ADC</td>
<td>Rd, Rr</td>
<td>Add with Carry two Registers</td>
<td>Rd ← (Rd + Rr) + C</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>ADDW</td>
<td>Rd,K</td>
<td>Add Immediate to Word</td>
<td>Rd ← Rd + Imm</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SUB</td>
<td>Rd, Rr</td>
<td>Subtract two Registers</td>
<td>Rd ← Rd - Rr</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SUBK</td>
<td>Rd, K</td>
<td>Subtract Constant from Register</td>
<td>Rd ← Rd - K</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SBC</td>
<td>Rd, K</td>
<td>Subtract with Carry two Registers</td>
<td>Rd ← Rd - Rr - C</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SBCI</td>
<td>Rd, K</td>
<td>Subtract with Carry Constant from Rng.</td>
<td>Rd ← Rd - Rr - K</td>
<td>Z, C, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>AND</td>
<td>Rd, Rr</td>
<td>Logical AND Registers</td>
<td>Rd ← Rd &amp; Rr</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>ANDI</td>
<td>Rd, K</td>
<td>Logical AND Register and Constant</td>
<td>Rd ← Rd &amp; K</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>OR</td>
<td>Rd, Rr</td>
<td>Logical OR Registers</td>
<td>Rd ← Rd</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>ORI</td>
<td>Rd, K</td>
<td>Logical OR Register and Constant</td>
<td>Rd ← Rd</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>COM</td>
<td>Rd</td>
<td>One's Complement</td>
<td>Rd ← 0xFF - Rd</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>NEG</td>
<td>Rd</td>
<td>Two's Complement</td>
<td>Rd ← 0x00 - Rd</td>
<td>Z, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SBR</td>
<td>R16, K</td>
<td>Set Bits(n) in Register</td>
<td>Rd ← K</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>CLR</td>
<td>R16, K</td>
<td>Clear Bits(n) in Register</td>
<td>Rd ← Rd</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>INC</td>
<td>Rd</td>
<td>Increment</td>
<td>Rd ← Rd + 1</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>DEC</td>
<td>Rd</td>
<td>Decrement</td>
<td>Rd ← Rd - 1</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>TST</td>
<td>Rd</td>
<td>Test for Zero or Neg.</td>
<td>Rd ← Rd</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>CLRC</td>
<td>Rd</td>
<td>Clear Register</td>
<td>Rd ← 0</td>
<td>Z, N, V</td>
<td>1</td>
</tr>
<tr>
<td>SET</td>
<td>Rd</td>
<td>Set Register</td>
<td>Rd ← 0xFF</td>
<td>None</td>
<td>1</td>
</tr>
<tr>
<td>MUL</td>
<td>Rd, Rr</td>
<td>Multiply Unsigned</td>
<td>Rd1 * Rd2 ← Rd1 * Rd2</td>
<td>Z, C</td>
<td>2</td>
</tr>
<tr>
<td>MULS</td>
<td>Rd, Rr</td>
<td>Multiply Signed with Unsigned</td>
<td>Rd1 * Rd2 ← Rd1 * Rd2</td>
<td>Z, C</td>
<td>2</td>
</tr>
<tr>
<td>FMUL</td>
<td>Rd, Rr</td>
<td>Fractional Multiply Unsigned</td>
<td>Rd1 * Rd2 ← Rd1 * Rd2</td>
<td>Z, C</td>
<td>2</td>
</tr>
<tr>
<td>FMULS</td>
<td>Rd, Rr</td>
<td>Fractional Multiply Signed with Unsigned</td>
<td>Rd1 * Rd2 ← Rd1 * Rd2</td>
<td>Z, C</td>
<td>2</td>
</tr>
</tbody>
</table>

## BRANCH INSTRUCTIONS

<table>
<thead>
<tr>
<th>Mnemonics</th>
<th>Operands</th>
<th>Description</th>
<th>Operation</th>
<th>Flags</th>
<th>#Clocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rjmp</td>
<td>k</td>
<td>Relative Jump</td>
<td>PC ← PC + k + 1</td>
<td>None</td>
<td>2</td>
</tr>
<tr>
<td>JUMP</td>
<td>k</td>
<td>Indirect Jump to (Z)</td>
<td>PC ← Z</td>
<td>None</td>
<td>2</td>
</tr>
<tr>
<td>RCALL</td>
<td>k</td>
<td>Relative Subroutine Call</td>
<td>PC ← PC + k + 1</td>
<td>None</td>
<td>3</td>
</tr>
<tr>
<td>ICALL</td>
<td>k</td>
<td>Indirect Call to (Z)</td>
<td>PC ← Z</td>
<td>None</td>
<td>3</td>
</tr>
<tr>
<td>RET</td>
<td></td>
<td>Subroutine Return</td>
<td>PC ← STACK</td>
<td>None</td>
<td>4</td>
</tr>
<tr>
<td>RETI</td>
<td></td>
<td>Interrupt Return</td>
<td>PC ← STACK</td>
<td>None</td>
<td>4</td>
</tr>
<tr>
<td>CPSE</td>
<td>Rd, Rr</td>
<td>Compare, Skip If Equal</td>
<td>if (Rd = Rr) then PC ← PC + 2 or 3</td>
<td>None</td>
<td>1/2/3</td>
</tr>
<tr>
<td>CF</td>
<td>Rd, Rr</td>
<td>Compare</td>
<td>Rd ← Rd</td>
<td>Z, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>CP</td>
<td>Rd, Rr</td>
<td>Compare with Carry</td>
<td>Rd ← Rd</td>
<td>Z, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>CPI</td>
<td>Rd, K</td>
<td>Compare Register with Immediate</td>
<td>Rd ← K</td>
<td>Z, N, V, H</td>
<td>1</td>
</tr>
<tr>
<td>SBRC</td>
<td>Rr, Rr</td>
<td>Skip if Bit in Register Cleared</td>
<td>if (Rr)bc(0) then PC ← PC + 2 or 3</td>
<td>None</td>
<td>1/2/3</td>
</tr>
<tr>
<td>SBSR</td>
<td>Rr, Rr</td>
<td>Skip if Bit in Register Set</td>
<td>if (Rr)bc(1) then PC ← PC + 2 or 3</td>
<td>None</td>
<td>1/2/3</td>
</tr>
<tr>
<td>SBCP</td>
<td>Rr, Rr</td>
<td>Skip if Bit in I/O Register Cleared</td>
<td>if (Rr)pc(0) then PC ← PC + 2 or 3</td>
<td>None</td>
<td>1/2/3</td>
</tr>
<tr>
<td>SBSP</td>
<td>Rr, Rr</td>
<td>Skip if Bit in I/O Register is Set</td>
<td>if (Rr)pc(1) then PC ← PC + 2 or 3</td>
<td>None</td>
<td>1/2/3</td>
</tr>
<tr>
<td>BREQ</td>
<td>k</td>
<td>Branch if Equal</td>
<td>if (Z = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRNE</td>
<td>k</td>
<td>Branch if Not Equal</td>
<td>if (Z = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRCS</td>
<td>k</td>
<td>Branch if Carry Set</td>
<td>if (C = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRCC</td>
<td>k</td>
<td>Branch if Carry Cleared</td>
<td>if (C = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRHS</td>
<td>k</td>
<td>Branch if Same or Higher</td>
<td>if (C = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRLO</td>
<td>k</td>
<td>Branch if Lower</td>
<td>if (C = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRMI</td>
<td>k</td>
<td>Branch if Minus</td>
<td>if (N = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRPL</td>
<td>k</td>
<td>Branch if Plus</td>
<td>if (N = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRGE</td>
<td>k</td>
<td>Branch if Greater or Equal, Signed</td>
<td>if (N = V = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRLT</td>
<td>k</td>
<td>Branch if Less Than, Signed</td>
<td>if (N = V = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRHS</td>
<td>k</td>
<td>Branch if Half Carry Flag Set</td>
<td>if (N = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRHC</td>
<td>k</td>
<td>Branch if Half Carry Cleared</td>
<td>if (N = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRST</td>
<td>k</td>
<td>Branch if T Flag Set</td>
<td>if (T = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRTR</td>
<td>k</td>
<td>Branch if T Flag Cleared</td>
<td>if (T = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRVS</td>
<td>k</td>
<td>Branch if Overflow Flag is Set</td>
<td>if (V = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRVC</td>
<td>k</td>
<td>Branch if Overflow Flag is Cleared</td>
<td>if (V = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRE</td>
<td>k</td>
<td>Branch if Interrupt Enabled</td>
<td>if (I = 1) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
<tr>
<td>BRID</td>
<td>k</td>
<td>Branch if Interrupt Disabled</td>
<td>if (I = 0) then PC ← PC + k + 1</td>
<td>None</td>
<td>1/2</td>
</tr>
</tbody>
</table>

## DATA TRANSFER INSTRUCTIONS

<table>
<thead>
<tr>
<th>Mnemonics</th>
<th>Operands</th>
<th>Description</th>
<th>Operation</th>
<th>Flags</th>
<th>#Clocks</th>
</tr>
</thead>
</table>

ATmega8535(L)
<table>
<thead>
<tr>
<th>Mnemonics</th>
<th>Operands</th>
<th>Description</th>
<th>Operation</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOV</td>
<td>Rd, Rr</td>
<td>Move Between Registers</td>
<td>Rd ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>MOVL</td>
<td>Rd, Rr</td>
<td>Copy Register Word</td>
<td>Rd ← Rd + Rr</td>
<td>None</td>
</tr>
<tr>
<td>LDI</td>
<td>Rd, K</td>
<td>Load Immediate</td>
<td>Rd ← K</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, X</td>
<td>Load Indirect</td>
<td>Rs ← (X)</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, X+</td>
<td>Load Indirect Post-Inc.</td>
<td>Rs ← (X)+</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, X-</td>
<td>Load Indirect Pre-Dec.</td>
<td>X ← X - 1, Rd ← X</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Y</td>
<td>Load Indirect</td>
<td>Rd ← (Y)</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Y+</td>
<td>Load Indirect Post-Inc.</td>
<td>Y ← Y + 1, Rd ← Y</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Y-</td>
<td>Load Indirect Pre-Dec.</td>
<td>Y ← Y - 1, Rd ← Y</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Rq</td>
<td>Load Indirect with Displacement</td>
<td>Rd ← (Y + q)</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Z</td>
<td>Load Indirect</td>
<td>Rd ← (Z)</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Z+</td>
<td>Load Indirect Post-Inc.</td>
<td>Rd ← (Z)+</td>
<td>None</td>
</tr>
<tr>
<td>LD</td>
<td>Rd, Z-</td>
<td>Load Indirect Pre-Dec.</td>
<td>Z ← Z - 1, Rd ← Z</td>
<td>None</td>
</tr>
<tr>
<td>LDD</td>
<td>Rd, Rq</td>
<td>Load Indirect with Displacement</td>
<td>Rs ← (Z + q)</td>
<td>None</td>
</tr>
<tr>
<td>LDD</td>
<td>Rd, k</td>
<td>Load Direct from SRAM</td>
<td>Rs ← (k)</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>X, Rr</td>
<td>Store Indirect</td>
<td>(X) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>X, Rr</td>
<td>Store Indirect and Post-Inc.</td>
<td>(X) ← Rr, X ← X + 1</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>X, Rr</td>
<td>Store Indirect and Pre-Dec.</td>
<td>X ← X - 1, (X) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>Y, Rr</td>
<td>Store Indirect</td>
<td>(Y) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>Y, Rr</td>
<td>Store Indirect and Post-Inc.</td>
<td>(Y) ← Rr, Y ← Y + 1</td>
<td>None</td>
</tr>
<tr>
<td>ST</td>
<td>Y, Rr</td>
<td>Store Indirect and Pre-Dec.</td>
<td>Y ← Y - 1, (Y) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>STD</td>
<td>Yq,Rr</td>
<td>Store Indirect with Displacement</td>
<td>(Y + q) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>STD</td>
<td>Z, Rr</td>
<td>Store Indirect</td>
<td>(Z) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>STD</td>
<td>Z, Rr</td>
<td>Store Indirect and Post-Inc.</td>
<td>(Z) ← Rr, Z ← Z + 1</td>
<td>None</td>
</tr>
<tr>
<td>STD</td>
<td>Z, Rr</td>
<td>Store Indirect and Pre-Dec.</td>
<td>Z ← Z - 1, (Z) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>STS</td>
<td>s, Rr</td>
<td>Store Direct to SRAM</td>
<td>(s) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>LPM</td>
<td>Rd, Z</td>
<td>Load Program Memory</td>
<td>R ← (Z)</td>
<td>None</td>
</tr>
<tr>
<td>LPM</td>
<td>Rd, Z+</td>
<td>Load Program Memory</td>
<td>R ← (Z)+</td>
<td>None</td>
</tr>
<tr>
<td>LPM</td>
<td>Rd, Z-</td>
<td>Load Program Memory</td>
<td>R ← (Z)-</td>
<td>None</td>
</tr>
<tr>
<td>LPM</td>
<td>Rd, Z</td>
<td>Load Program Memory and Post-Inc.</td>
<td>R ← (Z), Z ← Z + 1</td>
<td>None</td>
</tr>
<tr>
<td>SPH</td>
<td>Rd, Z</td>
<td>Store Program Memory</td>
<td>(Z) ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>IN</td>
<td>Rd, P</td>
<td>In Port</td>
<td>Rd ← P</td>
<td>None</td>
</tr>
<tr>
<td>OUT</td>
<td>P, Rr</td>
<td>Out Port</td>
<td>P ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>PUSH</td>
<td>Rr, P</td>
<td>Push Register on Stack</td>
<td>STACK ← Rr</td>
<td>None</td>
</tr>
<tr>
<td>POP</td>
<td>Rd, P</td>
<td>Pop Register from Stack</td>
<td>Rd ← STACK</td>
<td>None</td>
</tr>
</tbody>
</table>

### Bit and Test Instructions

| BIS       | P, s    | Set Bit in I/O Register | I/P + 1 | None  |
| CSI       | P, s    | Clear Bit in I/O Register | I/P + 0 | None  |
| LSL       | Rd      | Logical Shift Left | R ← Rl| None  |
| LSR       | Rd      | Logical Shift Right | R ← Rl| None  |
| ROL       | Rd      | Rotate Left Through Carry | R ← Rl| None  |
| ROR       | Rd      | Rotate Right Through Carry | R ← Rl| None  |
| ASR       | Rd      | Arithmetic Shift Right | R ← Rl| None  |
| SWAP      | Rs, Rs  | Swap Registers | R ← Rl| None  |
| BSET      | s, x    | Set Flag | S/E ← 1 | None  |
| BCLR      | s, x    | Clear Flag | S/E ← 0 | None  |
| BST       | Rd, x   | Bit Store from Register to T | T ← R| None  |
| BLD       | Rd, b   | Bit Load from 1 to Register | R ← b | None  |
| SEC       | C       | Set Carry | C ← 1 | None  |
| CLC       | C       | Clear Carry | C ← 0 | None  |
| SEN       | N       | Set Negative Flag | N ← 0 | None  |
| CLN       | N       | Clear Negative Flag | N ← 0 | None  |
| SEZ       | Z       | Set Zero Flag | Z ← 0 | None  |
| CLZ       | Z       | Clear Zero Flag | Z ← 0 | None  |
| SEI       | I       | Global Interrupt Enable | I ← -1 | None  |
| CLI       | I       | Global Interrupt Disable | I ← 1 | None  |
| SES       | B       | Set Signed Test Flag | B ← 1 | None  |
| CLS       | B       | Clear Signed Test Flag | B ← 0 | None  |
| SEV       | Y       | Set Two's Complement Overflow | Y ← 1 | None  |
| CLV       | Y       | Clear Two's Complement Overflow | Y ← 0 | None  |
| SET       | T ← S/E  | Set T in S/E | T ← 1 | None  |
| CLT       | T       | Clear T in S/E | T ← 0 | None  |
| SEH       | H       | Set Half Carry Flag in S/E | H ← 1 | None  |
| CLH       | H       | Clear Half Carry Flag in S/E | H ← 0 | None  |

### MCU Control Instructions

| NOP       | No Operation | None  |

---

2502KS-AVR-10/06

D - 24
<table>
<thead>
<tr>
<th>Mnemonics</th>
<th>Operands</th>
<th>Description</th>
<th>Operation</th>
<th>Flags</th>
<th>#Clocks</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLEEP</td>
<td>Sleep</td>
<td>(see specific descr. for Sleep function)</td>
<td>None</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>WDR</td>
<td>Watchdog Reset</td>
<td>(see specific descr. for WDT/Timer)</td>
<td>None</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>BREAK</td>
<td>Break</td>
<td>For On-chip Debug Only</td>
<td>None</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>
## Ordering Information

<table>
<thead>
<tr>
<th>Speed (MHz)</th>
<th>Power Supply</th>
<th>Ordering Code</th>
<th>Package(^{(1)})</th>
<th>Operation Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>2.7 - 5.5V</td>
<td>ATmega8535L-8AC</td>
<td>44A</td>
<td>Commercial (0°C to 70°C)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8PC</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8JC</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8MC</td>
<td>44M1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8AI</td>
<td>44A</td>
<td>Industrial (-40°C to 85°C)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8PI</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8JI</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8MI</td>
<td>44M1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8AU(^{(2)})</td>
<td>44A</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8PU(^{(2)})</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8JU(^{(2)})</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535L-8MU(^{(2)})</td>
<td>44M1</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4.5 - 5.5V</td>
<td>ATmega8535-16AC</td>
<td>44A</td>
<td>Commercial (0°C to 70°C)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16PC</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16JC</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16MC</td>
<td>44M1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16AI</td>
<td>44A</td>
<td>Industrial (-40°C to 85°C)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16PI</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16JI</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16MI</td>
<td>44M1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16AU(^{(2)})</td>
<td>44A</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16PU(^{(2)})</td>
<td>40P6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16JU(^{(2)})</td>
<td>44J</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATmega8535-16MU(^{(2)})</td>
<td>44M1</td>
<td></td>
</tr>
</tbody>
</table>

Note:  
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.

2. Pb-free packaging alternative, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.

### Package Type

<table>
<thead>
<tr>
<th>Package Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>44A</td>
<td>44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)</td>
</tr>
<tr>
<td>40P6</td>
<td>40-pin, 0.600&quot; Wide, Plastic Dual Inline Package (PDIP)</td>
</tr>
<tr>
<td>44J</td>
<td>44-lead, Plastic J-leaded Chip Carrier (PLCC)</td>
</tr>
<tr>
<td>44M1-A</td>
<td>44-pad, 7 x 7 x 1.0 mm body, lead pitch 0.50 mm, Quad Flat No-Lead/Micro Lead Frame Package (QFN/MLF)</td>
</tr>
</tbody>
</table>
Packaging Information

44A

Notes:
1. This package conforms to JEDEC reference MS-026, Variation ACB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>MIN</th>
<th>NOM</th>
<th>MAX</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>–</td>
<td>–</td>
<td>1.20</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>0.05</td>
<td>–</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>0.05</td>
<td>1.00</td>
<td>1.05</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>11.75</td>
<td>12.00</td>
<td>12.25</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>9.90</td>
<td>10.00</td>
<td>10.10</td>
<td>Note 2</td>
</tr>
<tr>
<td>E</td>
<td>11.75</td>
<td>12.00</td>
<td>12.25</td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>9.90</td>
<td>10.00</td>
<td>10.10</td>
<td>Note 2</td>
</tr>
<tr>
<td>B</td>
<td>0.30</td>
<td>–</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0.09</td>
<td>–</td>
<td>0.20</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>0.45</td>
<td>–</td>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>0.80 TYP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ATmega8535(L)

40P6

COMMON DIMENSIONS
(Unit of Measure = mm)

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>MIN</th>
<th>NOM</th>
<th>MAX</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>–</td>
<td>–</td>
<td>4.030</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>0.381</td>
<td>–</td>
<td>–</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>52.070</td>
<td>–</td>
<td>52.570</td>
<td>Note 2</td>
</tr>
<tr>
<td>E</td>
<td>15.240</td>
<td>–</td>
<td>15.875</td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>13.462</td>
<td>–</td>
<td>13.970</td>
<td>Note 2</td>
</tr>
<tr>
<td>B</td>
<td>0.356</td>
<td>–</td>
<td>0.559</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>1.041</td>
<td>–</td>
<td>1.651</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>3.048</td>
<td>–</td>
<td>3.556</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>0.203</td>
<td>–</td>
<td>0.381</td>
<td></td>
</tr>
<tr>
<td>eB</td>
<td>15.494</td>
<td>–</td>
<td>17.526</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>–</td>
<td>–</td>
<td>2.540 TYP</td>
<td></td>
</tr>
</tbody>
</table>

Notes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold flash or protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

09/28/01

AIMEL
2325 Orchard Parkway
San Jose, CA 95131

TITLE
40P6, 40-lead (0.600/15.24 mm Wide) Plastic Dual Inline Package (PDIP)

DRAWING NO.
40P6

REV.
B

2502KS-AVR-10/06

D - 28
Notes:
1. This package conforms to JEDEC reference MS-016, Variation AC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is \(0.010\)\((0.254 \text{ mm})\) per side. Dimension D1 and E1 include mold mismatch and are measured at the extreme material condition at the upper or lower parting line.
3. Lead coplanarity is \(0.004^\circ\) \((0.102 \text{ mm})\) maximum.

COMMON DIMENSIONS
(Unit of Measure - mm)

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>MIN</th>
<th>NOM</th>
<th>MAX</th>
<th>NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4.191</td>
<td>-</td>
<td>4.572</td>
<td></td>
</tr>
<tr>
<td>A1</td>
<td>2.286</td>
<td>-</td>
<td>3.048</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>0.508</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>17.399</td>
<td>-</td>
<td>17.653</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>16.510</td>
<td>-</td>
<td>16.662</td>
<td>Note 2</td>
</tr>
<tr>
<td>E</td>
<td>17.399</td>
<td>-</td>
<td>17.653</td>
<td></td>
</tr>
<tr>
<td>E1</td>
<td>16.510</td>
<td>-</td>
<td>16.662</td>
<td>Note 2</td>
</tr>
<tr>
<td>B</td>
<td>0.660</td>
<td>-</td>
<td>0.813</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>0.330</td>
<td>-</td>
<td>0.533</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>1.270 TYP</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10/04/01
44M1-A

D - 30
Errata

ATmega8535
Rev. A and B

The revision letter refer to the device revision.

- First Analog Comparator conversion may be delayed
- Asynchronous Oscillator does not stop in Power-down

1. First Analog Comparator conversion may be delayed
   If the device is powered by a slow rising $V_{DD}$, the first Analog Comparator conversion will take longer than expected on some devices.
   **Problem Fix/Workaround**
   When the device has been powered or reset, disable then enable the Analog Comparator before the first conversion.

2. Asynchronous Oscillator does not stop in Power-down
   The asynchronous oscillator does not stop when entering Power-down mode. This leads to higher power consumption than expected.
   **Problem Fix/Workaround**
   Manually disable the asynchronous timer before entering Power-down.
Datasheet Revision History

Please note that the referring page numbers in this section are referring to this document. The referring revision in this section are referring to the document revision.

Changes from Rev. 2502J- 08/06 to Rev. 2502K- 10/06
1. Updated TOP/BOTTOM description for all Timer/Counters Fast PWM mode.
2. Updated “Errata” on page 18.

Changes from Rev. 2502I- 06/06 to Rev. 2502J- 08/06

Changes from Rev. 2502H- 04/06 to Rev. 2502I- 06/06
1. Updated code example “USART Initialization” on page 150.

Changes from Rev. 2502G- 04/05 to Rev. 2502H- 04/06
1. Added “Resources” on page 6.
2. Updated Table 7 on page 29, Table 17 on page 42 and Table 111 on page 258.
4. Updated note in “Bit Rate Generator Unit” on page 180.

Changes from Rev. 2502F- 06/04 to Rev. 2502G- 04/05
1. Removed “Preliminary” and TBD’s.
2. Updated Table 37 on page 69 and Table 113 on page 261.
3. Updated “Electrical Characteristics” on page 255.

Changes from Rev. 2502E-12/03 to Rev. 2502G-06/04
1. MLF-package alternative changed to “Quad Flat No-Lead/Micro Lead Frame Package QFN/MLF”.

Changes from Rev. 2502E-12/03 to Rev. 2502F-06/04
1. Updated “Reset Characteristics” on page 37.
2. Updated SPH in “Stack Pointer” on page 12.
3. Updated C code in “USART Initialization” on page 150.
4. Updated “Errata” on page 18.

Changes from Rev. 2502D-09/03 to Rev. 2502E-12/03
1. Updated “Calibrated Internal RC Oscillator” on page 29.
Changes from Rev. 2502C-04/03 to Rev. 2502D-09/03

1. Removed “Advance Information” and some TBD's from the datasheet.
2. Added note to “Pinout ATmega8535” on page 2.
3. Updated “Reset Characteristics” on page 37.
4. Updated “Absolute Maximum Ratings” and “DC Characteristics” in “Electrical Characteristics” on page 255.
5. Updated Table 111 on page 258.
6. Updated “ADC Characteristics” on page 263.
7. Updated “ATmega8535 Typical Characteristics” on page 266.
8. Removed CALL and JMP instructions from code examples and “Instruction Set Summary” on page 10.

Changes from Rev. 2502B-09/02 to Rev. 2502C-04/03

1. Updated “Packaging Information” on page 14.
2. Updated Figure 1 on page 2, Figure 84 on page 179, Figure 85 on page 185, Figure 87 on page 191, Figure 98 on page 207.
3. Added the section “EEPROM Write During Power-down Sleep Mode” on page 22.
4. Removed the references to the application notes “Multi-purpose Oscillator” and “32 kHz Crystal Oscillator”, which do not exist.
5. Updated code examples on page 44.
7. Renamed Port D pin ICP to ICP1. See “Alternate Functions of Port D” on page 64.
8. Added information about PWM symmetry for Timer 0 on page 79 and Timer 2 on page 126.
9. Updated Table 68 on page 169, Table 75 on page 190, Table 76 on page 193, Table 77 on page 196, Table 108 on page 253, Table 113 on page 261.
10. Updated description on “Bit 5 – TWSTA: TWI START Condition Bit” on page 182.
11. Updated the description in “Filling the Temporary Buffer (Page Loading)” and “Performing a Page Write” on page 231.
12. Removed the section description in “SPI Serial Programming Characteristics” on page 254.
14. Updated “ADC Characteristics” on page 263.
14. Updated “Register Summary” on page 8.
15. Various Timer 1 corrections.
16. Added WD_FUSE period in Table 108 on page 253.

Changes from Rev. 2502A-06/02 to Rev. 2502B-09/02
1. Changed the Endurance on the Flash to 10,000 Write/Erase Cycles.
Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2500

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chanterie
BP 70692
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-87-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN ATTEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATEL'S WEB SITE, ATTEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATTEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATTEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Atmel’s products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

© 2006 Atmel Corporation. All rights reserved. Atmel®, logo and combinations thereof, Everywhere You Are®, AVR®, and others are the trademarks or registered trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.