

Code Examples for Dios Class Microcontrollers

i2c Slave Example	1
DS1994 Timer Example	6
DS1994 Memery Example	8
DS1920 Interface Example	9
Reset Reason Example	10
Timer1 as External Clock	10
IRQ Wakeup Example	11
bintodec Example	12
irkeypad Example	12
irread Example	12
UART Example	13
Hardware PWM Example	13
sleep/wakeup/watchdog Example	14
arrayget/arrayset Example	15
AtoD Example	15
data Example	15
Auto Dialer Example	16

i2c Slave Example

This is a real neat I2c program. It will turn any Dios Chip into a I2c Slave and allow other microcontollers to changes its global variables.

Its all handled in the background so you dont have to think about it. Its only 325 bytes long.

Slave Example

```
-----
'This is a neat little I2c slave program. With just 325 bytes it lets any microcontroler
' that can talk I2c read and write to its global variables
'I2c Ports are: sda=Port11 scl=Port12
'Make sure you hold them high with a 10k resistor.
```

```
func main()
  setupI2cslave(72) 'Sets address
  global testglob as integer

'Do notheing
loop:
  print "Hello world. Our test global varb=",testglob
```

```
pause 500
goto loop
```

```
endfunc
```

```
'=====
'This routines does all the work
' It handles all the SSP requests and keeps track of which packet byte has
' been recieved or transmitted
'=====
```

```
startirqasm SSP
```

```
I2cADDR equ BYTE0
I2cGIDX equ BYTE1
I2cGVALH equ BYTE2
I2cGVALL equ BYTE3
I2cCOUNT equ BYTE4
```

```
'Check to see if stop condition
IF_ SSPSTAT,4 = 1
    clrf I2cCOUNT
    goto irqdone
ENDIF_
```

```
'-----
'Master sends data
```

```
IF_ SSPSTAT,2 = 0
    IF_ SSPSTAT,5 = 1
```

```
    IF_ I2cCOUNT = 0
        movff SSPBUF,I2cGIDX
        bsf I2cGIDX,7
        incf I2cCOUNT,f
        goto irqdone
    ENDIF_
```

```
    IF_ I2cCOUNT = 1
        movff SSPBUF,I2cGVALH
        incf I2cCOUNT,f
        goto irqdone
    ENDIF_
```

```
    IF_ I2cCOUNT = 2
        movff SSPBUF,I2cGVALL
        movff I2cGIDX,VARBIDX
        movff I2cGVALL,VARBSAVEL
        movff I2cGVALH,VARBSAVEH
        call writearbvalue
        clrf I2cCOUNT
        goto irqdone
    ENDIF_
```

```
    movff SSPBUF,BYTE1
ELSE_
    movff SSPBUF,I2cADDR
```

```
ENDIF_  
ENDIF_
```

```
'-----  
' Master requests data. Here we respond  
'If its write and data  
IF_ SSPSTAT,2 = 1  
  IF_ I2cCOUNT = 1  
    movff I2cGIDX,VARBIDX  
    call readvarbvalue  
    movff VARBSAVEL,I2cGVALL  
    movff VARBSAVEH,I2cGVALH  
    movff I2cGVALH,SSPBUF  
    incf I2cCOUNT,f  
    bsf SSPCON1,4  
    goto irqdone  
  ENDIF_  
  
  IF_ I2cCOUNT = 2  
    movff I2cGVALL,SSPBUF  
    bsf SSPCON1,4  
    goto irqdone  
  ENDIF_  
  
ENDIF_  
  
irqdone  
  
endirqasm
```

```
'-----  
'This sets up the I2c hardware  
'-----  
func setupI2cslave(addr)  
  SSPSTAT=0  
  SSPSTAT.bit(7) =0 ' slewrate 400Khz  
  SSPSTAT.bit(6) =0 ' Disable Bus inputs  
  'Rest of the bits are inputs for dtecting various states  
  
  SSPADD = addr  
  
  SSPCON1 = %0110 'Sets Slave I2c mode 7bit no start and stop IRQ  
  SSPCON1.bit(5)=1 'Enables SSP Port  
  SSPCON1.bit(4)=1 'Release clock  
  
  BYTE4=0 'Set counter  
  
'IRQ Stuff  
  irqSSPstart()  
  irqperfstart()  
  irqglobalstart()
```

```
endfunc
```

Master Example

'This program will use I2c to send and retrieve data to a remote

' slaves Global variables

```
func main()
```

```
  dim x,y
```

```
  const sda 0
```

```
  const scl 1
```

```
loop:
```

'Use this to change the slaves global variables

```
  for x = 650 to 2550
```

```
    break "send ",x," to global 0"
```

```
    setremoteglobal(sda,scl,72,0,x)
```

```
  next
```

```
goto loop
```

```
'-----
```

'Use this to retrieve data from slaves global variables

```
  for x = 0 to 3
```

```
    break "get value of global ",x
```

```
    y=getremoteglob(sda,scl,72,x)
```

```
    print "Value = ",y
```

```
  next
```

```
goto loop
```

```
endfunc
```

```
'=====
```

'Sets the remote global variable

```
'=====
```

```
func setremoteglobal(sda,scl,addr,gvarb,dat)
```

```
  dim value
```

```
'Send start
```

```
'break "Send start"
```

```
I2c_start(sda,scl)
```

```
'break "Send address"
```

```
'Control byte
```

```
I2c_sendbyte(sda,scl,addr) 'EEProm write
```

```
I2c_getack(sda,scl)
```

```
'break "Send data"
```

```
'send index
```

```
I2c_sendbyte(sda,scl,gvarb)
```

```
I2c_getack(sda,scl)
```

```
'send value high
I2c_sendbyte(sda,scl,dat.byte(1))
I2c_getack(sda,scl)
```

```
'send value high
I2c_sendbyte(sda,scl,dat.byte(0))
I2c_getack(sda,scl)
```

```
I2c_stop(sda,scl)
```

```
endfunc
```

```
'=====
'Retrieves the value of a global variable
'=====
```

```
func getremoteglob(sda,scl,addr,gvarb)
  dim value as integer
```

```
'Send start
'break "Send start"
I2c_start(sda,scl)
```

```
'break "Send address"
'Control byte
I2c_sendbyte(sda,scl,addr) 'EEProm write
I2c_getack(sda,scl)
```

```
'break "Send data"
'send data
I2c_sendbyte(sda,scl,gvarb)
I2c_getack(sda,scl)
```

```
'break "Send start2"
I2c_start(sda,scl)
```

```
'break "Send address read"
'Control byte
addr.bit(0)=1
I2c_sendbyte(sda,scl,addr) 'EEProm read
I2c_getack(sda,scl)
```

```
'break "Get byte 1"
value.byte(1) = I2c_getbyte(sda,scl)
'print "value=",value
I2c_sendack(sda,scl)
```

```
'break "Get byte 2"
value.byte(0) = I2c_getbyte(sda,scl)
'print "value=",value
```

```
skip:
'break "send stop"
I2c_stop(sda,scl)
```

exit value

endfunc

include \lib\DiosI2c.lib

DS1994 Timer Example

iButton DS1994 Timer Example

'1994 IButton timer Demo 1

'Shows how to read the timer

'Note this example does not use the ROM match feature so only 1 DS1994 per bus

func main()

dim indat1 as integer

const OWPort 8

clear

DS1994writebyte(8,513,16) 'Write byte

'Clear counter

DS1994writeword(8,514,0) 'Write word

DS1994writeword(8,516,0) 'Write word

'Read 16 bits of seconds

loop:

indat1=DS1994readword(8,515) 'read real time bytes 1 & 2

print indat1, " Seconds"

pause 1000

goto loop

endfunc

'-----
'Write a byte to the iButton memory
'-----

func DS1994writebyte(owport,address,tdata)

'Write to scratch pad

OWreset(owport)

OWsendbyte(owport,\$CC)

OWsendbyte(owport,\$0F)

OWsendbyte(owport,address.byte(0))

OWsendbyte(owport,address.byte(1))

```
OWsendbyte(owport,tdata)
```

```
'Copy Scratch pad to mem
```

```
OWreset(owport)
```

```
OWsendbyte(owport,$CC)
```

```
OWsendbyte(owport,$55)
```

```
OWsendbyte(owport,address.byte(0))
```

```
OWsendbyte(owport,address.byte(1))
```

```
OWsendbyte(owport,address.byte(0) & 31)
```

```
OWreset(owport)
```

```
endfunc
```

```
'-----
```

```
'Write a word to the iButton memory
```

```
'-----
```

```
func DS1994writeword(owport,address,tdata)
```

```
'Write to scratch pad
```

```
OWreset(owport)
```

```
OWsendbyte(owport,$CC)
```

```
OWsendbyte(owport,$0F)
```

```
OWsendbyte(owport,address.byte(0))
```

```
OWsendbyte(owport,address.byte(1))
```

```
OWsendbyte(owport,tdata.byte(0))
```

```
OWsendbyte(owport,tdata.byte(1))
```

```
'Copy Scratch pad to mem
```

```
OWreset(owport)
```

```
OWsendbyte(owport,$CC)
```

```
OWsendbyte(owport,$55)
```

```
OWsendbyte(owport,address.byte(0))
```

```
OWsendbyte(owport,address.byte(1))
```

```
OWsendbyte(owport,address.byte(0) & 31 + 1)
```

```
OWreset(owport)
```

```
endfunc
```

```
'-----
```

```
'Read a byte from the iButton memory
```

```
'-----
```

```
func DS1994readbyte(owport,address)
```

```
dim tret as integer
```

```
OWreset(owport)
```

```
OWsendbyte(owport,$CC)
```

```
OWsendbyte(owport,$F0)
```

```
OWsendbyte(owport,address.byte(0))
```

```
OWsendbyte(owport,address.byte(1))
```

```
tret = OWreadbyte(owport)
```

```
OWreset(owport)
```

```
exit tret
```

```
endfunc
```

```

'-----
'Read a word from the iButton memory
'-----
func DS1994readword(owport,address)
  dim tret as integer

  OWreset(owport)
  OWsendbyte(owport,$CC)
  OWsendbyte(owport,$F0)
  OWsendbyte(owport,address.byte(0))
  OWsendbyte(owport,address.byte(1))
  tret.byte(0) = OWreadbyte(owport)
  tret.byte(1) = OWreadbyte(owport)
  OWreset(owport)
  exit tret
endfunc

include \lib\Dios1Wire.lib

```

Be very careful what you write to the control register (\$201 or 513). The lower three bits (0-2) set the write protect options of the 1994. If these are set (by performing "copy scratchpad" three times) AND one of the alarms goes off, You will be permanently locked out of the 1994!

DS1994 Memery Example

iButton DS1994 Code Examples

```

'1994 IButton Demo
'Shows how to read and write to memory
'The DS1994 has 512 bytes of memory
'Note this example does not use the ROM match feature so only 1 DS1994 per bus
func main()

```

```

  dim indat1 as integer

```

```

  DS1994writebyte(8,221,4) 'Write byte

```

```

  indat1=DS1994readbyte(8,221)
  print indat1

```

```

endfunc

```

```

'-----
'Write a byte to the iButton memory
'-----
func DS1994writebyte(owport,address,tdata)

```

```

'Write to scratch pad
OWreset(owport)
OWsendbyte(owport,$CC)
OWsendbyte(owport,$0F)
OWsendbyte(owport,address.byte(0))
OWsendbyte(owport,address.byte(1))
OWsendbyte(owport,tdata)

'Copy Scratch pad to mem
OWreset(owport)
OWsendbyte(owport,$CC)
OWsendbyte(owport,$55)
OWsendbyte(owport,address.byte(0))
OWsendbyte(owport,address.byte(1))
OWsendbyte(owport,address.byte(0) & 31)

OWreset(owport)
endfunc

```

```

'-----
'Read a byte from the iButton memory
'-----
func DS1994readbyte(owport,address)
dim tret as integer

OWreset(owport)
OWsendbyte(owport,$CC)
OWsendbyte(owport,$F0)
OWsendbyte(owport,address.byte(0))
OWsendbyte(owport,address.byte(1))
tret = OWreadbyte(owport)
OWreset(owport)
exit tret
endfunc

include \lib\Dios1Wire.lib

```

DS1920 Interface Example

iButton DS1920 interface code example

Remember to hold port high with 5k resistor.

```

'1920 IButton Demo 1
'Note we can use the 1820 library and force it into Parasite mode.
'All IButtons operate in Parasite mode
func main()

dim CelTempF as float 'To hold temperature
dim FarTempF as float 'To hold temperature

```

global ROM(9) as string 'Used to hold ROM code

clear

'First Load Rom Code into string variable

'Note the code for IButtons is read from right to left

'Every 2 digits is a hex code. Use the Conversion utility

'To convert to decimal

OWloadaddr(16,44,36,59,0,8,0,125,ROM)

'Note you can use the following routine to read and display iButton ROM values

'OWreadrom(8,temp1)

'OWprintaddr(temp1)

'Here we just read the temp and display it

loop:

CelTempF = DS1820readtemp(8,3,ROM)

FarTempF = CelsiustoF(CelTempF)

print {4.1} CelTempF," ",FarTempF

goto loop

endfunc

include \lib\Dios1820.lib

Reset Reason Example

Shows how to determine the reason for reset.

'Example of detecting reason for reset.

func main()

print "Resetflag=",RESETFLAG

if RESETFLAG = 0 then print "Power Up"

if RESETFLAG = 1 then print "Reset"

if RESETFLAG = 2 then print "Brown Out Reset"

if RESETFLAG = 3 then print "Software Reset"

if RESETFLAG = 4 then print "Watch Dog Reset"

if RESETFLAG = 5 then print "sleep reset"

endfunc

Timer1 as External Clock

Shows how to set timer1 as an external clock with a 32Khz Xtal.

'This example puts the Dios to sleep then wakes it up with the
'external 32khz oscillator.

'-----

```

'Demonstrate putting Dios to sleep
'It will wake up every 15 seconds
'You must connect a 32.768 Xtal across Port 14 and 15
' You must also connect a 47pf cap from port 14 to gnd and
' port 15 and gnd.
func main()
  pause 100

  print "reset "

'Only Timer 1 can do this
  timer1setvalue 0 'Clear internal counters
  timer1mode16bit '16 bit mode
  timer1prescale 3'Sets prescale
  timer1oscon 'Turn on oscillator
  timer1syncon 'Async
  timer1sourceP15 'External clock
  timer1on 'Start Timer

'Start IRQs to wake up chip
  irqglobalstart
  irqperfstart
  irqTMR1start

again:

  print "Go to sleep"
  sleep
  print "wake up"

  goto again

endfunc

```

IRQ Wakeup Example

Shows how to use a button to wakeup a dios from sleep mode using an IRQ.

```

'Use button on port 7 to wake up Dios
'This program puts the Dios to sleep.
func main()

  irqINT0clear
  irqINT0start
  irqperfstart
  irqglobalstart

loop:
  print "Going to Sleep....."
  sleep
  print "Wake Up"

  goto loop

```

```
endfunc
```

bintodec Example

Shows how to use the bintodec command to convert a variable to its ASCII parts.

```
'bintodec example
func main()
  dim value
  dim v10000,v1000,v100,v10,v1

  value = 5000

  'Break value down into digits
  bintodec value,v10000,v1000,v100,v10,v1

  'Display digits
  print v10000," ",v1000," ",v100," ",v10," ",v1

endfunc
```

irkeypad Example

Shows how to read the keypad from a SONY IR remote.

```
'Dios Program to test the IRkeypad function
func main()
  dim number

loop:
  number=IRkeypad(3,11,4)
  print
  print number
  goto loop

endfunc

include \lib\DiosIR.lib
```

irread Example

Shows how to read data from a SONY IR remote control with the irread command.

```
'Dios Program to test the IRread function
func main()
  dim cmd,device

'Use this to power Vishay IR module
'Place pin 1 in pin 10
' 2 in pin 11
```

```
' 3 in pin 12
  output 11,12
  low 11
  high 12

loop:
  cmd = IRread(10,1000)
  if cmd =0 then
    goto loop
  endif
  print IRdevice," ",cmd
  goto loop

endfunc

include \lib\DiosIR.lib
```

UART Example

Shows how to use the built-in UART with the hserin hserout commands.

```
'UART Example
func main()
  dim inchar

  hsersetup baud,HBAUD9600,start,txon

mainloop:
  hserin mainloop,inchar
  hserout inchar
  print "Recieved ",inchar

'Now go back and do it all again
  goto mainloop

endfunc
```

Hardware PWM Example

Shows how to setup the hardware PWM using the hpwm command.

```
'Use button on IO port 4 and 5 to change the hardware PWM
generator
'Hardware PWM example
'PWM 1 = port 13
func main()
  dim x,drc,prd

  PWMinit(1)
  drc = 1
  prd = 255

  print
```

```

print "Course = ",drc," prd=",prd

again:
  PWMcourse(drc)
  PWMperiod(prd)
  PWM1duty(prd/2)

loop:

  if ioport(4) = 1 then
    while ioport(4)= 1
      wend
      drc = drc + 1
      if drc > 2 then drc = 0
      print "Course = ",drc," prd=",prd
      goto again
    endif

  if ioport(5) = 1 then
    while ioport(5)= 1
      wend
      prd = prd - 1
      if prd < 1 then prd = 10
      drc = 1
      print "Course = ",drc," prd=",prd
      goto again
    endif

  goto loop
endfunc

include \lib\DiosHWPWM.lib

```

sleep/wakeup/watchdog Example

Shows how to put the Dios to sleep and wake it up every 2 seconds with the watchdog timer.

```

'sleep example
func main()
  print "Reset"

  watchdogon

loop:
  print "Going to sleep for two seconds"
  sleep
  print "We are awake"
  goto loop

endfunc

```

arrayget/arrayset Example

shows how to access normal variables as though they were arrays.

'This demo shows the arrayget and arrayset commands. They are used to access multiple non array integer variables.

'Demo of arrayget and arrayset commands

```
func main()
  dim a,b,c,d,e,z,x

  for x = 0 to 4
    arrayset a,x,x * 10
  next

  for x = 0 to 4
    arrayget a,x,z
    print z," ";
  next

endfunc
```

AtoD Example

Shows how to set up a couple of the 10-bit AtoD ports and display the results.

'Dios Program to demonstrate AtoD library

```
func main()
  dim a,b,c

  AtoDinit(4) 'Init 3 Analog ports

loop:
  a=AtoD(0)
  b=AtoD(1)
  c=AtoD(3)
  print {04} a," ",b," ",c
  pause 500
  goto loop

endfunc

include \lib\DiosAtoD.lib
```

data Example

Shows how to use the data command to setup default data in the on board eeprom. There is 256 bytes available with this data space. It is non volatile so it will stay put even if the power is removed.

'Demo of data command
'The Data Command writes default data to the internal EEPROM space

```
func main()  
  dim a,b  
  
  data 100,200,600,1,2,3,4  
  
  pause 100  
  
  eeread 0,a  
  print a  
  eeread 1,a  
  print a
```

'Note that the third item in the data statement writes 2 bytes to eeprom

```
  eeread 2,a  
  eeread 3,b  
  print a * 256 + b  
  
  eeread 4,a  
  print a  
  eeread 5,a  
  print a  
  eeread 6,a  
  print a  
  eeread 7,a  
  print a  
  
endfunc
```

Auto Dialer Example

This auto dialer program will show how the Dios can control a USRobotics Modem. In this example we use it to place votes for American Idol contestants.

```
'-----  
'American Idol Dialer  
func main()  
  dim inchar as integer  
  global phone(15) as string  
  global phone2(15) as string  
  global indat(100) as string  
  
  dim counts,stat,calls  
  
  clear  
  
  hsetup baud,HBAUD9600,start,txon  
  print "Standby ...."  
  pause 1000  
  
'Phone numbers of top two contestants  
  phone = "18664365705"  
  phone2 = "18664365703"
```

```

dialagain:
  pause 500
  counts = 0
  if calls.bit(1)=1 then
    hserout "atX3dt",phone,13
  else
    hserout "atX3dt",phone2,13
  endif

  calls = calls + 1
  indat = 0

'-----
loop:
'Time out loop
  if counts > 8000 then
    print "Timeout"
    goto dialagain
  endif

  counts = counts + 1
  pause 1
  hserin loop,inchar
  debug inchar

  if inchar = 10 or inchar = 13 then
    stat = STRinstr(indat,"NO CARRIER")
    if stat 1000 then goto dialagain
    stat = STRinstr(indat,"BUSY")
    if stat 1000 then goto dialagain
  endif

  indat = *+inchar
  goto loop

endfunc

include \lib\DiosString.lib

```