







## Features

-  10/100BaseT Ethernet port
-  Single-channel RS232 port (DB9)
-  Power input through DB9
-  Optional power-over-Ethernet
-  Compact (90x48x25mm)
-  DIN rail, wall mounting plates included



## About

The DS1100 is a compact BASIC-programmable controller targeting cost-sensitive serial-over-IP and serial control applications.

Built around the same T2000 ASIC as our EM500 “MiniMo” module, the DS1100 offers a single-channel RS232 port. Device power can be supplied via its power jack, pin9 of the DB9 (serial port) connector, or the optional power-over-Ethernet board.

There are three LEDs on the device’s front: green and red main status LEDs, and a yellow Ethernet link LED.

Each DS1100 is supplied with the DIN rail and wall mounting plates.

The device comes preloaded with a fully functional serial-over-IP application.

## Specifications

- Superior upgrade to the DS203 device.
- Based on the high-performance T2000 IC.
- 10/100BaseT, auto-MDIX Ethernet port.
- Optional power-over-Ethernet.
- Single-channel RS232 port on DB9M connector:
  - TX, RX, RTS, CTS, DTR, and DSR lines;
  - Baudrates of up to 115200bps;
  - None/even/odd/mark/space parity modes;
  - 7/8 bits/character modes;
  - Optional RTS/CTS flow control;
  - “12V” power input on pin 9 of the DB9.
- 512KB flash memory for firmware and application.
- 200 bytes of EEPROM space for data storage.
- Three LEDs:
  - Green and red main status LEDs;
  - Yellow Ethernet link LED.
- Power: 12VDC nominal (min. 9V, max. 18V).
- Dimensions: 90x48x25mm.
- Operating temperature range: -5 ~ 70 C.

## Specifications (continued)

- Firmware is upgradeable through the serial port or network (including cold upgrade firmware uploads through the network).
- CE- and FCC-certified.
- Included accessories:
  - DIN rail mounting plate
  - Wall mounting plate and two screws
- Optional Accessories:
  - 12V/0.5A adaptor: APR-P0011 (US), APR-P0012 (EU), APR-P0013 (UK)
  - WAS-1499 straight Ethernet cable\*
  - WAS-P0004(B) DB9M-to-DB9F serial cable (device-to-PC)
  - WAS-P0005(B) DB9F-to-DB9F serial cable (device-to-device)

\*For this device can be used as crossover cable too

## Platform Objects

- Sock — socket comms (up to 16 UDP, TCP, and HTTP sessions).
- Net — controls Ethernet port.
- Ser — in charge of serial channels.
- Io — handles I/O lines, ports, and interrupts.
- Stor — provides access to the EEPROM.
- Romfile — facilitates access to resource files (fixed data).
- Pppoe — accesses the Internet over an ADSL modem.
- Ppp — accesses the Internet over a serial modem (GPRS, etc.).
- Pat — “plays” patterns on green and red status LEDs.
- Button — monitors the setup button.
- Sys — in charge of general device functionality.

## Tibbo Integrated Development Environment (TIDE)

It literally takes one key — F5 — to compile your Tibbo BASIC project, upload it onto the target, and run it with full debugging capabilities. You don't need any special debugging hardware. Just connect your DS1100 to the LAN and debug right through the network.

The screenshot displays the TIDE interface with several callout boxes explaining key features:

- Project browser:** See all objects, system calls, etc.
- Call stack:** Trace the flow of program execution
- Output pane:** See compile errors and debug trace messages
- Code editor:** Syntax highlighting, zoom, code hinting, completion
- Debug toolbar:** Start, pause, execute line by line
- Watch pane:** Read and modify variable values
- Status bar:** See target state in real-time

```
end sub
public sub html_init(buf_alloc as no_yes)
  dim f as byte
  'TELNET/HTTP
  if buf_alloc = YES then
    for f=SOCK_HTTP to SOCK_HTTP+NUM_HTTP_PORTS-1
      sock.num=f
      sock.txbufirq(6)
      sock.rxbufirq(1)
      sock.varbufirq(1)
    next f
  else
    for f=SOCK_HTTP to SOCK_HTTP+NUM_HTTP_PORTS-1
      sock.num=f
      sock.txbufirq(0)
      sock.rxbufirq(0)
      sock.varbufirq(0)
    next f
  end if
  sys.bufalloc
  'TELNET/HTTP
  for f=SOCK_HTTP to SOCK_HTTP+NUM_HTTP_PORTS-1
    sock.num=f
```

Name	Type	Value
sock.num	byte	2 (&h02)
sock.txfree	word	495 (&h01ef)
sys.freebufpages	byte	36 (&h24)
sock.tflen	word	0 (&h0000)
sock.state	pl_sock_state_simple	0 - PL_SSTS_IDLE
is_login_status	no_yes	0 - NO
sock.num	byte	2 (&h02)