WAVECREST ® Corporation
A Technologies Company

Product Description — DSM-16S
Dual 8:1 Solid State Digital Switching Matrix

Introduction

Dual 8-input Switch Matrix – A 19” x 22” x 3.5” rack mountable, dual 8-input binary switch matrix with a band-width of over 1.5 GHz. This unit is intended to extend the input capability of the DTS-207X from 2 to 16 channels. The matrix can be controlled manually from the front panel or remotely via the RS-232C port at the back of the DTS. Remote control of the DSM-16S via GPIB commands to the DTS makes integration into an automated environment fast with no special hardware or software required. The DSM-16S is designed to be used as a 1 of 8 matrix to the DTS channel input (1 of 8 to channel 1, and 1 of 8 to channel 2 — See Figure 1). The DSM-16S includes an RS232C cable for connecting to the DTS-207X Series of Digital Time Measurement products.

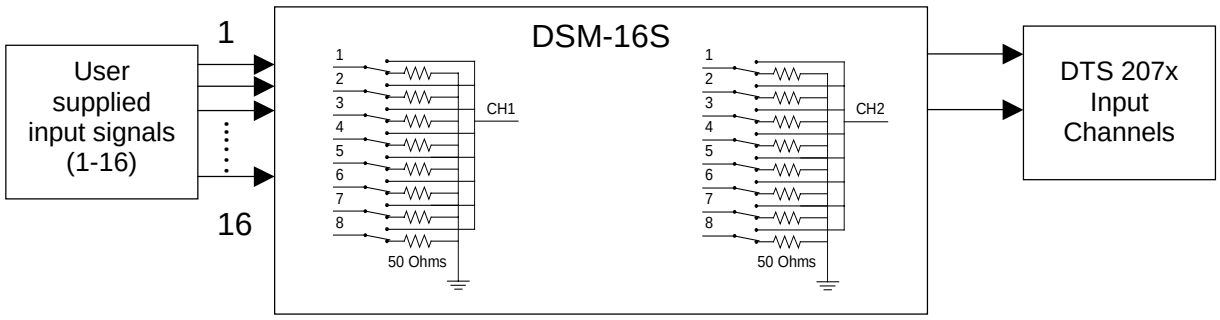

Figure 1

Performance Specifications

Frequency range	DC - 1.5 GHz
VSWR	1.2:1 typical, 1.5:1 maximum
Insertion loss	2.0 dB typical, 2.75 dB maximum (1.5 GHz)
Isolation	>35 dB typical, 30 dB minimum
Skew to any path	±25 ps maximum
Impedance	50 Ω nominal
Unused inputs	Terminated 50 Ω to ground
Matrix switching time	<15 ms
Relay control	RS232C
Front panel inputs	± 1.3 VAC maximum
Minimum relay life	100 million switches per channel (Maximum input voltage must not exceed ± 1.3 VAC)

Power Requirements

Voltage requirements	90 to 260 VAC @ 50-60 Hz
Input current	0.125 A maximum
Fuses	(2) 0.25 A, 250 VAC, 5 x 20 mm

Temperature Requirements

Storage	0 - 70° C
Operating	5 - 40° C

Humidity

Storage	85%, Non-condensing
Operating	85%, Non-condensing, up to 31° C

Intended Use Of Equipment

The DSM-16S should be used only for its intended purpose as outlined in this product description. To avoid possible injury, the DSM-16S should not be operated with the top cover or other panels removed. Refer installation and maintenance to qualified service personnel. To avoid explosion, do not operate the DSM-16S in or near an atmosphere of explosive gases. It is essential to maintain the protective earth ground through the grounding connector of the power cord. A loss of the protective ground can cause electrical shock. The DSM-16S is a Class 1 LED product. When the DSM-16S is installed into a rack enclosure, the rear power switch and power cord must be accessible for proper power disconnect or a proper power disconnect must be provided.

Installation of the DSM-16S

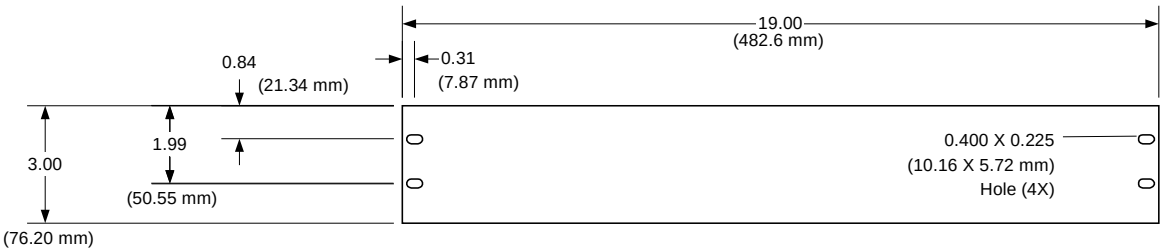

Figure 1 - Rack Mount Dimensions

1. Place the DSM under the DTS unit.
2. Connect Channel 1 of the DSM to Channel 1 of the DTS.
3. Connect Channel 2 of the DSM to Channel 2 of the DTS.
4. Connect user inputs to appropriate DSM inputs.
5. Connect RS232C cable to RS232C port on the back of the DSM. Connect the other end of the cable to the back of the DTS-207X.
6. Connect power cord to unit and to AC power source.

User installation complete.

The DSM-16S front panel connections are only intended to be connected to signals where applied voltages will not exceed 1.3 $V_{pk\text{-}pk}$ AC (Installation Category I).

Fuse Replacement

The IEC plug-in at the back of the DSM-16S provides the input connection for the AC power cord. A small compartment on the IEC plug houses two 0.25 A/250 V fuses (5x20 mm). The manufacturer's part number is Wickmann IEC 19195-035. To gain access to the fuses, remove the power to the DSM and disconnect the power cord. With a small screwdriver, pry open the fuse compartment on the IEC plug. Remove the fuse and install the new fuse prior to closing the compartment and reinstalling the power cord.

Cleaning

The outside of the DSM-16S should be cleaned with Isopropyl alcohol.

Remote Programming of the DSM-16S

Refer to the GPIB Programming Guide, Section 8, Channel Commands, included with your DTS-207x for DSM-16S control commands.

RS232 Interface

The interface operates at 9600 Baud, No Parity, 8 Bits and 1 Stop Bit. The basic command set is:

D - Disable manual switches
E - Enable manual switches
R - Read switches (4 bytes)
V - Read version number (3 bytes)
Wxy - write switch, x = bank, y = switch (All ASCII)
X - return char in output buffer, ? if empty

Hardware handshaking is not supported. On the read commands, handshaking is accomplished by initiating the command (R or V) then sending an "X" each time you are ready for another byte.

Sample Program

```
#include <windows.h>
#include <stdio.h>

// All functions return 0 if successful
BOOL OpnDSM16 ( int FrontPanelOn );
BOOL SetDSM16 ( int Bank, int Channel );
BOOL GetDSM16 ( int *Bank1Channel, int *Bank2Channel );

HANDLE  hCom;

int main ( void )
  {
  int pass, chan, bank, retn[ 2 ];

  // Open the ComPort, leave the front panel switches enabled
  if ( OpnDSM16 ( 1 ) )                          goto Error;
  // Scan through all valid channels on both banks 10 times
  // Read back the current bank/channel to validate
  for ( pass = 1; pass <= 10; pass++ )
    for ( bank = 1; bank <= 2; bank++ )
      for ( chan = 1; chan <= 8; chan++ )
        {
        if ( SetDSM16 ( bank, chan ) )             goto Error;
        if ( GetDSM16 ( &retn[ 0 ], &retn[ 1 ] ) ) goto Error;
        if ( retn[ bank - 1 ] != chan )            goto Error;
        }
  printf ( "DSM16 passed test....\n" );
  return 0;
Error:
  printf ( "DSM16 failed test....\n" );
  return -1;
  }

BOOL OpnDSM16 ( int FrontPanelOn )
  {
  DCB           dcb;
  COMMTIMEOUTS  cto;
  double        vers;
  int           indx;
  DWORD         read;
  char          buff[ 4 ];

  // Configure an open serial port, see Windows documentation for details
  if ( ! BuildCommDCB ( "96,N,8,1", &dcb ) ) return -1;
  hCom = CreateFile ( "COM1:", GENERIC_READ | GENERIC_WRITE, 0, NULL,
CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL );
  if ( hCom == INVALID_HANDLE_VALUE ) return -1;
  cto.ReadIntervalTimeout         = 3;
  cto.ReadTotalTimeoutMultiplier  = 5;
  cto.WriteTotalTimeoutMultiplier = 5;
```

```
  cto.ReadTotalTimeoutConstant    = 50;
  cto.WriteTotalTimeoutConstant   = 50;
  if ( ! SetCommTimeouts ( hCom, &cto ) ) return -1;
  if ( ! SetupComm ( hCom, 2048, 2048 ) ) return -1;
  if ( ! SetCommState ( hCom, &dcb ) )    return -1;
  // First send an "v" to to obtain the version
  if ( ! WriteFile ( hCom, "v", 1, &read, NULL ) ) return -1;
  // Handshaking is accomplished by sending an "x"
  // and then reading each of the three individual bytes
  for ( indx = 0; indx < 3; indx++ )
    {
    if ( ! WriteFile ( hCom, "x", 1, &read, NULL ) ) return -1;
    if ( ! ReadFile ( hCom, &buff[ indx ], 1, &read, NULL ) ) return -1;
    }
  // Convert the version number and validate that it is at least 1.3
  buff[ 3 ] = 0;
  sscanf ( buff, "%lf", &vers );
  if ( vers < 1.3 ) return -1;
  // Now enable/disable the front panel switches
  switch ( FrontPanelOn )
    {
    case 0:
      if ( ! WriteFile ( hCom, "d", 1, &read, NULL ) ) return -1;
      break;
    default:
      if ( ! WriteFile ( hCom, "e", 1, &read, NULL ) ) return -1;
    }
  return 0;
  }

BOOL SetDSM16 ( int Bank, int Channel )
  {
  DWORD read;
  char  buff[ 4 ];

  // The write command is a "w" followed by the bank and chan
  sprintf ( buff, "w%c%c", Bank + '0',  Channel + '0' );
  return ( ! WriteFile ( hCom, buff, 3, &read, NULL ) );
  }

BOOL GetDSM16 ( int *Bank1Channel, int *Bank2Channel )
  {
  int   indx;
  DWORD read;
  char  buff[ 4 ];

  // First send an "r" to initiate the read
  if ( ! WriteFile ( hCom, "r", 1, &read, NULL ) ) return -1;
  // Handshaking is accomplished by sending an "x"
  // and then reading each of the four individual bytes
  for ( indx = 0; indx < 4; indx++ )
    {
    if ( ! WriteFile ( hCom, "x", 1, &read, NULL ) ) return -1;
    if ( ! ReadFile ( hCom, &buff[ indx ], 1, &read, NULL ) ) return -1;
    }
  // The four returned bytes are: '1', bank1channel, '2', bank2channel
  if ( buff[ 0 ] != '1' || buff[ 2 ] != '2' ) return -1;
  if ( buff[ 1 ] <  '1' || buff[ 1 ] >  '8' ||
       buff[ 3 ] <  '1' || buff[ 3 ] >  '8' ) return -1;
  // Place the validated values in their locations and return
  *Bank1Channel = buff[ 1 ] - '0';
  *Bank2Channel = buff[ 3 ] - '0';
  return 0;
  }
```