

## Communication with external devices

### Sound card

Requires data acquisition toolbox

Make sure that proper device is set in sound recorder. If using external signal, choose Microphone input.

- 1) Open device
- 2) Add channels
- 3) Configure property values
- 4) Acquire data
- 5) Close device

```
>> ai = analoginput('winsound');  
>> addchannel(ai,1:2);  
>> rate=44100; time=5;  
>> set(ai,'SampleRate',rate,'SamplesPerTrigger',rate*time);  
>> start(ai); data = getdata(ai); delete(ai);  
>> sound(data,rate)
```

External devices can be connected to the microphone input and that data can be captured.

Sound card acquires only AC signal.

If sound from a CD player is captured, make sure to select CD player as the source of sound.

```
>> !sndvol32 &
```

CD player can be started from Matlab:

```
>> !cdplayer &
```

Sound card can be used to output the signal, for example

```
>> ao = analogoutput('winsound'); % Use analogoutput function
>> addchannel(ao,1:2);
>> set(ao,'SampleRate',rate);
>> freq=500;
>> data = sin(linspace(0,2*pi*freq*time,rate*time));
>> putdata(ao,[data data])
>> start(ao)
>> delete(ao)
```

### Parallel port

Also needs Data Acquisition Toolbox

Use function *daqhwinfo* to display data acquisition hardware information: *out = daqhwinfo('adaptor')*

'adaptor' - The hardware driver adaptor name.

The supported adaptors are hpe1432, keithley, mcc, nidaq, parallel, and winsound. Example:

```
>> daqhwinfo('parallel')
ans =
    AdaptorDllName: 'C:\MATLAB\toolbox\daq\daq\private\mwparallel.dll'
    AdaptorDllVersion: 'Version 2.2 (R13) 28-Jun-2002'
    AdaptorName: 'parallel'
    BoardNames: {'PC Parallel Port Hardware'}
    InstalledBoardIds: {'LPT1'}
    ObjectConstructorName: {'" " 'digitalio('parallel','LPT1')'}
```

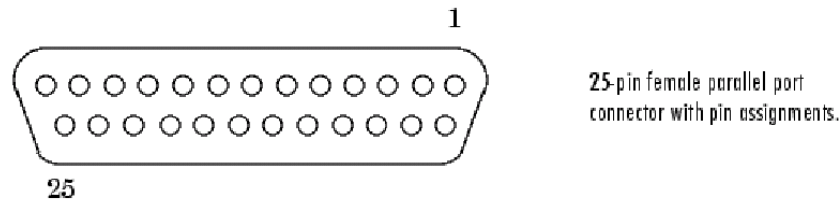
To use parallel port, need to open the port with *digitalio* function

```
DIO1 = digitalio('parallel','LPT1');
```

Next need to add lines to the IO device. For example:

```
in_lines = addline(DIO1, 0:7, 0, 'in');
```

The parallel port consists of eight data lines, four control lines, five status lines, and eight ground lines. In normal usage, the lines are controlled by the host computer software and the peripheral device following a protocol such as IEEE Standard 1284-1994. The protocol defines procedures for transferring data such as handshaking, returning status information, and so on. However, the toolbox uses the parallel port as a basic digital I/O device, and no protocol is needed. Therefore, you can use the port to input and output digital values just as you would with a typical DIO subsystem. To access the physical parallel port lines, most PCs come equipped with one 25-pin female connector, which is shown below.



The lines use TTL logic levels. A line is high (true or asserted) when it is a TTL high level, while a line is low (false or unasserted) when it is a TTL low level. The exceptions are lines 1, 11, 14, and 17, which are hardware inverted. The toolbox groups the 17 nonground lines into three separate ports. The port IDs and the associated pin numbers are given below.

Port ID	Pins	Description
0	2-9	Eight I/O lines, with pin 9 being the most significant bit (MSB).
1	10-13, and 15	Five input lines used for status
2	1, 14, 16, and 17	Four I/O lines used for control

Note that even though Port 0 lines can be used for data input and output, usually Matlab can not change value of the bit that is responsible for bidirectional data I/O. Still do not know how to solve this problem on XP. With W2k computers, program called “parmon” can be used to set bit 5 @ Port 2 to value 1.

Data is collected using function *getvalue*

```
dat = getvalue('parallel_input_handle');
```

Digital data can be output with *putvalue*

```
putvalue('parallel_output_handle',data);
```

or

```
putvalue(dio.Line(1:8),data)
```

Finish with clean up: `delete(dio)`

Consider as an example of using parallel port *compensator\_change.m* function.

### Serial port

- 1) Create or find serial object. Create: *serial* function. Finding: *instrfind* function.
- 2) Set up communication parameters. Use *set* function.
- 3) Open serial port w/ *fopen*.
- 4) I/O with *fwrite* or *fread*.
- 5) Close and delete port!

Examples: *tds.m* and *metex.m*

### Image capture

Requires image acquisition toolbox

- 1) Identify (select) video adapter. *imaqhwinfo* function
- 2) Identify (select) video mode and create appropriate object. *vid=videoinput('winvideo',adapter\_num);*
- 3) Get data either with *data=getsnapshot(vid);* or by using triggered events (see example below)
- 4) Stop video capture, delete video object

As examples consider *video\_select.m* and *waveguide1.m* files