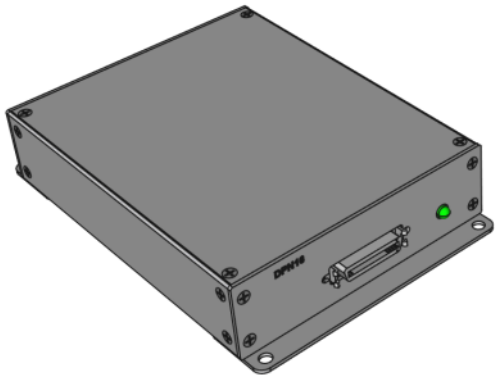
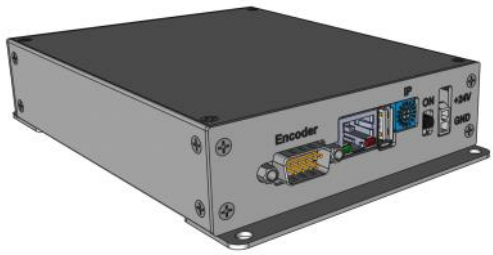
# DPN16 Driver

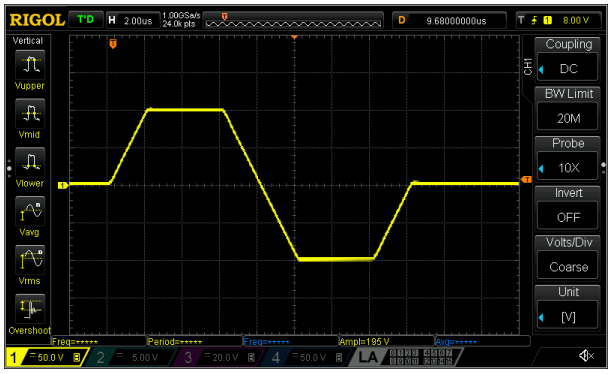




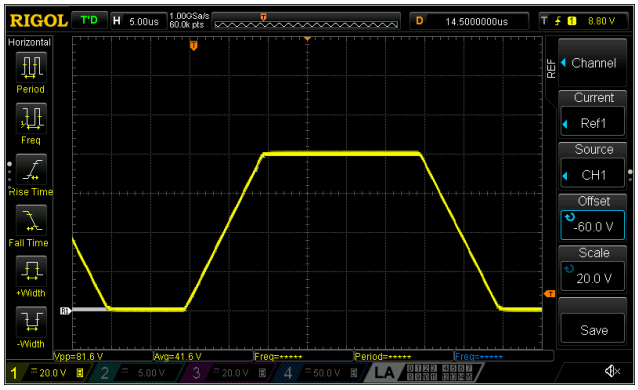

The DPN16 driver can output 16 independent high-voltage waveforms.
It can drive Dimatix DMC(Samba) cartridge and any piezoelectric actuators for inkjet or dispensor.

| | |
|---|---|
| waveform | • individually programmable 16 channel outputs<br>• voltage range : -100V ~ 100V<br>• voltage resolution : 0.05 V<br>(not measured, 12 bit DAC with 200V span)<br>• waveform length : max 500 microsecond<br>• waveform time resolution: 50 ns ~ 1 us<br>(variable depending on the waveform length) |
| spitting | • internal programmable frequency generator : 100 Hz ~ 50 kHz<br>• internal programmable drop counter : 15 bit count (1~32,768) |
| printing | • printing according to the position encoder signals<br>• 32 MBytes ring buffer for image data<br>• position encoder : RS422/RS485 level<br>• position encoder counter : 25 bit |
| communication | • 10/100 Mbps Fast Ethernet<br>• USB2.0 HighSpeed<br>• Software : C style DLL or C# class library DLL |
| power | • DC 24V<br>• standby current : 0.4 A |
| mechanical | • size 162 x 114 x 33 mm<br>• weight 480 g |

Waveform performance

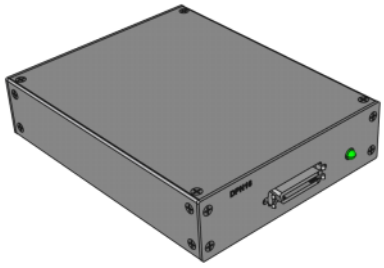



±100V waveform with Dimatix DMC cartridge (slew = 50V/µs, max slew rate = 100V/us)

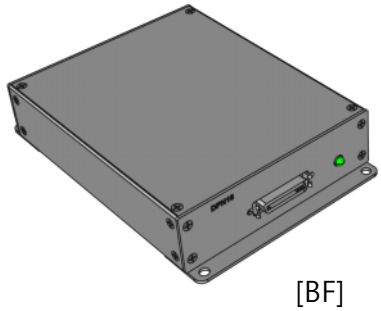



80V/10 µs slew rate with 2 nF

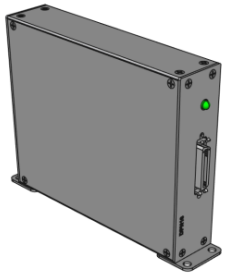
Attachment options



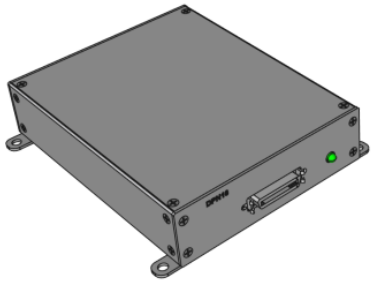
[NONE]



[BF]



[SF]



[BS]

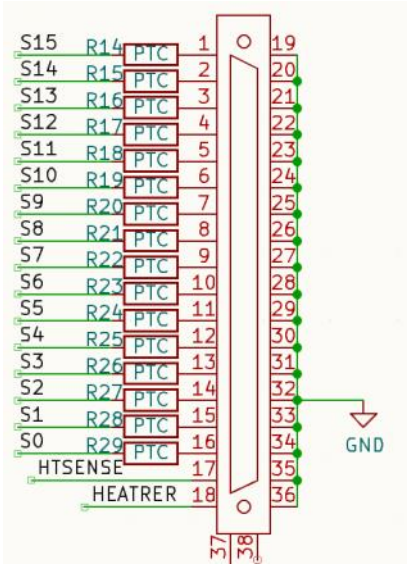
# DPN16 connection

1. Waveform output connector

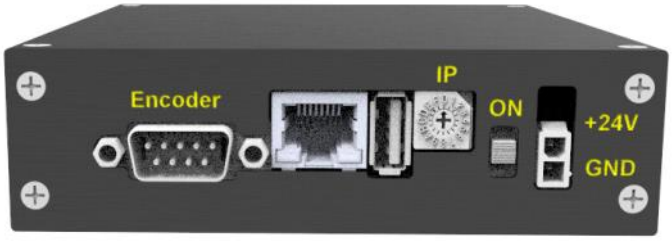
MDR36 pin connector (manufacturer : 3M)

driver side: 10236-55G3PC, mating cable : 10136-3000PE

| | |
|---|---|
| pair 1 | pin 1 : signal 15, pin 19 : GND |
| pair 2 | pin 2 : signal 14, pin 20 : GND |
| pair 3 | pin 3 : signal 13, pin 21 : GND |
| pair 4 | pin 4 : signal 12, pin 22 : GND |
| pair 5 | pin 5 : signal 11, pin 23 : GND |
| pair 6 | pin 6 : signal 10, pin 24 : GND |
| pair 7 | pin 7 : signal 9, pin 25 : GND |
| pair 8 | pin 8 : signal 8, pin 26 : GND |
| pair 9 | pin 9 : signal 7, pin 27 : GND |
| pair 10 | pin 10 : signal 6, pin 28 : GND |
| pair 11 | pin 11 : signal 5, pin 29 : GND |
| pair 12 | pin 12 : signal 4, pin 30 : GND |
| pair 13 | pin 13 : signal 3, pin 31 : GND |
| pair 14 | pin 14 : signal 2, pin 32 : GND |
| pair 15 | pin 15 : signal 1, pin 33 : GND |
| pair 16 | pin 16 : signal 0 , pin 34 : GND |
| pair 17 | pin 17 : signal thermistor, pin 35 : GND |
| pair 18 | pin 18 : signal heater, pin 36 : GND |




MISUMI assembled cable part No. : GRNET-HH-WA-36-1


2. Communication

2-1. Ethernet cable : 10/100 Mbps Fast Ethernet

rotary dip switch : 0 ~ 15

IP address = 192.168.0.100 + rotary dip switch value

(ex) rotary dip switch value = 2

IP address = 192.1668.0.102

2-2. USB : USB2.0

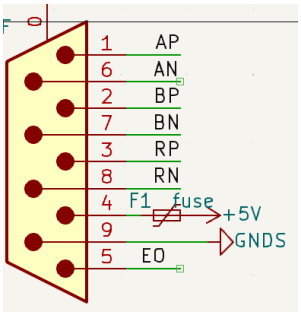
WinUSB device : no need to install USB device driver

The rotary dip switch can be used to distinguish one driver from multiple drivers.

3. Power connector : manufacturer : Molex  part No. 0039301020 (5569-2)

   pin 1 : GND,  pin2 : 24V DC

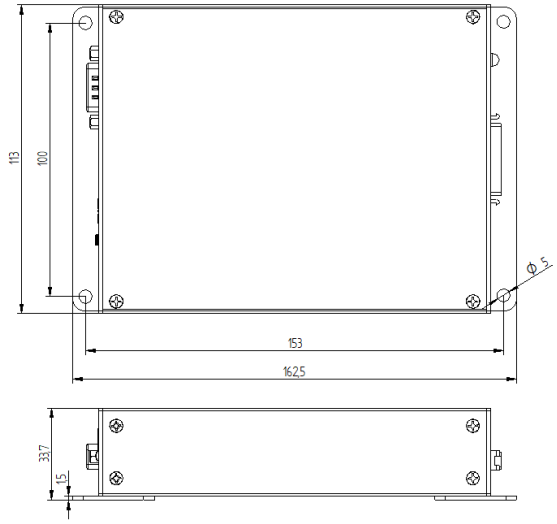

4. Encoder : DSUB 9 pin , driver side : male connector,  cable side : female connector

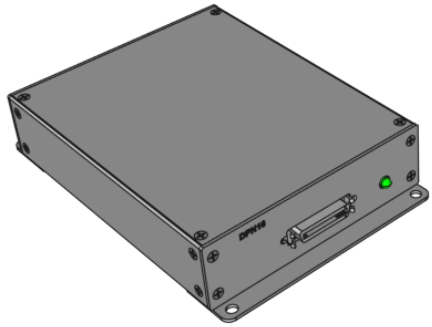



- encoder signal AP/AN, BP/BN
- 5V DC power output for encoder,  5V,  500 mA

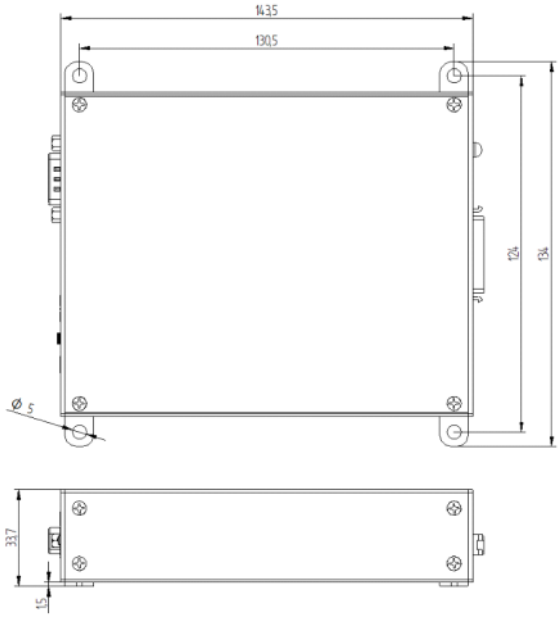
- E0 is not encoder signal.
  E0 is trigger output signal for stroboscope synchronization.

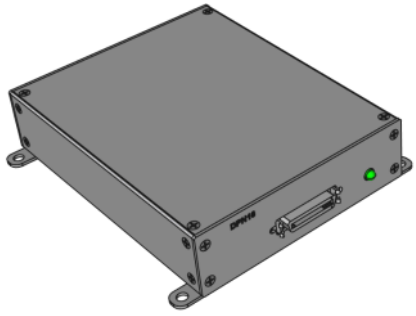
# DPN16 attachement



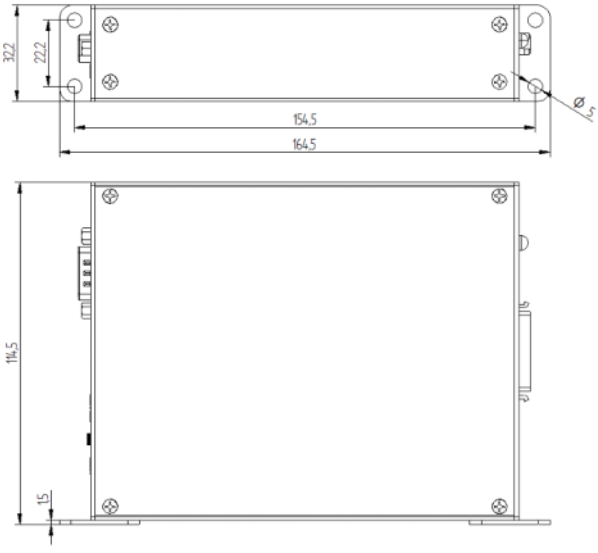



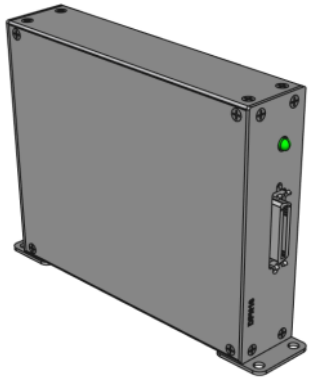
option: BF



option: BS



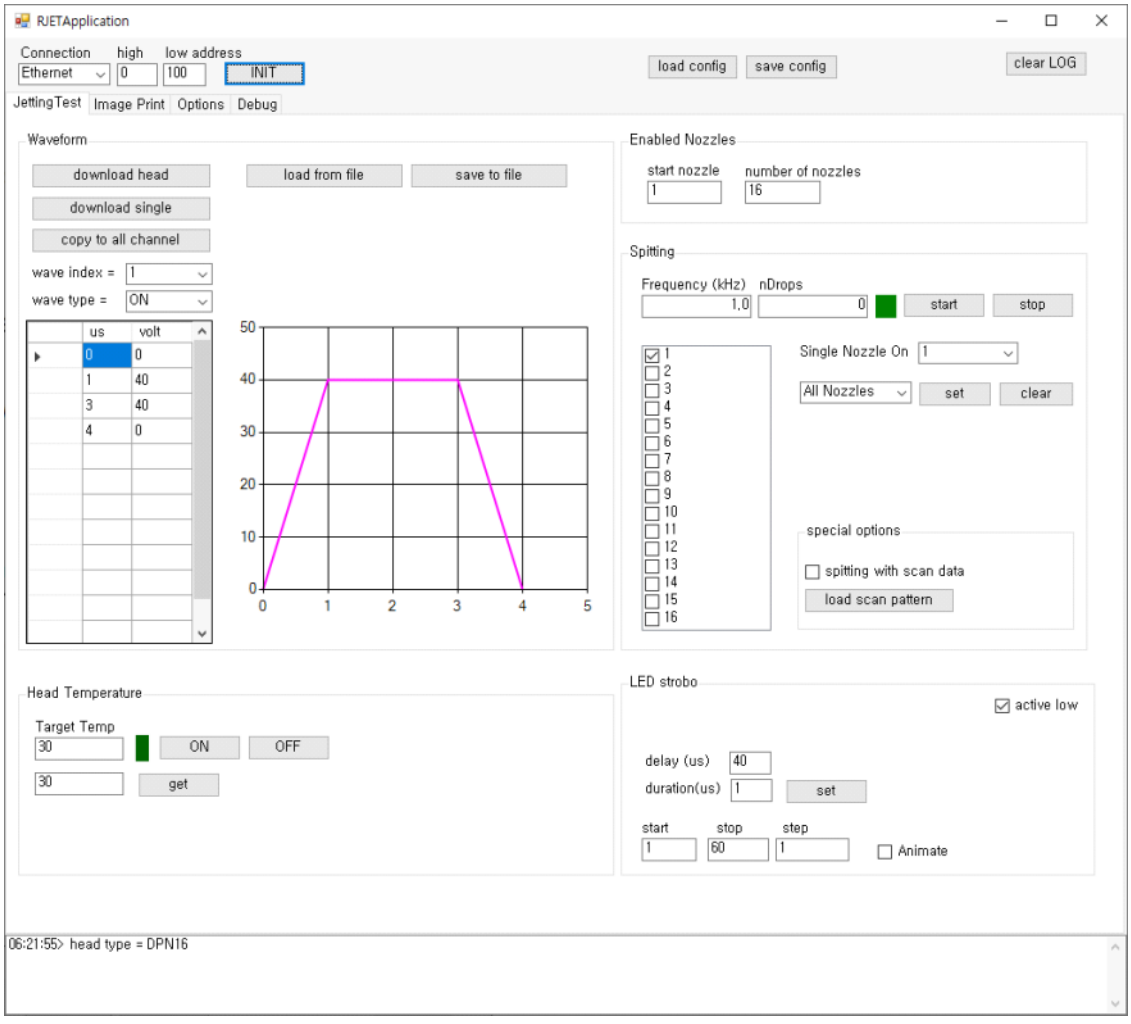
option: SF

# DPN16 TestApplication

1. Software installation

| | |
|---|---|
| RDApp4A.exe | test application execution file |
| RDriver.dll | DPN16 Driver class library |
| XCOM.dll<br>WinUSBNet.dll | RDriver.dll depends on this file |
| RDriverC.dll | C/C++ dll for MFC application |
| RDriverWrapperForC.dll | RDriverC.dll depends on this file |

2. TestApplication




2-1. Communication : Ethernet

a. choose 'Ethernet' option.
b. set high address = 0,  set low address = 100 + rotary dip switch value
   (ex) if rotary dip switch indicates 4, low address = 104, IP = 192.168.0.104
c. press the button "INIT"

2-2. Communication : USB

a. DPN16 Driver is a WinUSB device which does not require any USB device driver installation.
b. set high address = 0,  set low address = 100 + rotary dip switch value
c. press the button "INIT"

2-3. Setting waveforms

a. DPN16 has 16 channels. Each channel has two waveforms, ON and OFF.
b. On waveform can be used when the nozzle is selected to fire.
c. Off waveform can be used when the nozzle is not selected to fire.
   If you do not program, off waveforms are just 0V DC.
   Off waveform can be programmed and may used as tickle stimuli when the nozzle is not selected to fire.
d. set the waveform index and type and fill the waveform parameters.
e. press "DOWNLOAD" button.

2-4. Head Temperature

a. set Target temperature and press "ON" button.
b. get current head temperature by pression "GET" button
c. press "OFF" button to stop heating the print head.

2-5. Spitting

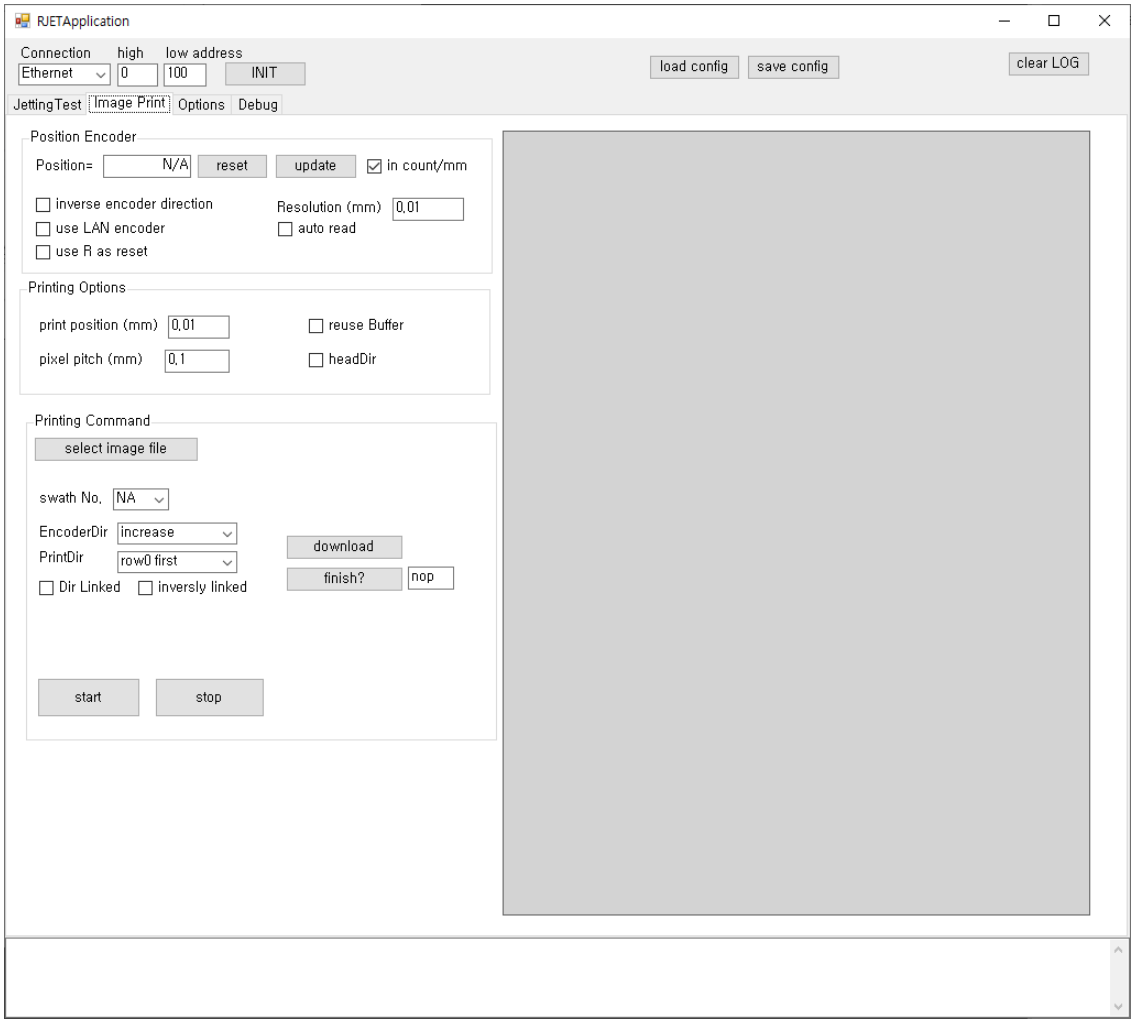
a. select nozzles to fire.
b. set spitting frequency.  0.1 kHz ~ 50 kHz
c. set count of spitting.  1 ~ 32767 counts.
   If you set count of spitting as 0, spitting will continue until you press "STOP" button.

2-6. Trigger out for LED stroboscope

a. set delay time and duration time of trigger output.
b. set polarity of trigger out.
c. Trigger signal will be available at S0 pin of encoder conector when spitting enabled.

2-7. Position encoder

a. Before printing, you should check the position encoder counter is properly updated.
b. "reset" button resets counter as 0.
c. "update" button reads encoder counter and displayed it.
d. You can change the direction(polarity).




2-8. Printing

a. select image file from disk.
   JPG, PNG, GIF, TIFF, BMP
b. press "download" button to write data into DPN16 driver.
c. reset encoder counter
d. press "start" button and move the carriage to generate encoder signals.
e. press "stop" button after carriage movement finished.

# DPN16 class library

1. DPN16 class library is written in C#.
   RDriver.dll : class library
   XCOM.dll, WinUSBNet.dll : RDriver.dll depends on these files

2. Usage

   ```
   using RDriver;
   wHeadDriver RD = new wHeadDriver();
   ```

   RD.ErrorMessage is a string for read message

3. Methods

   3-1. initalization

   ```
   bool init(int low,int high=0, bool useUSB = false, bool inverseNozzle = false)
   ```

   low = 100 + rotary dip switch value (0 ~ 15)
   IP address = 192.168.0.low

   ex) rotary dip switch =0, Ethernet
   init(100);

   3-2. position encoder

   bool setEncoderDir(bool inverse) : encoder direction setting, default=false
   bool resetEncoder() : reset encoder counter as 0
   int getEncoder() : get encoder counter

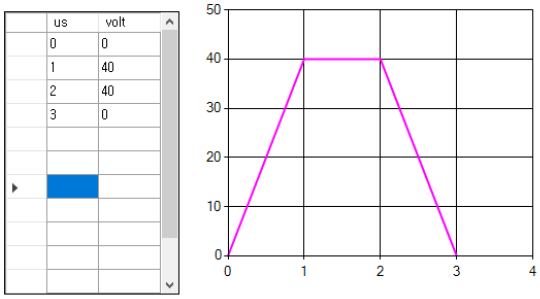
   3-3. waveform

   ```
   bool setWaveform(int channel, double[] CP, int nCP)
   ```

   channel : 0 ~ 15 : ON waveform index
   16 ~ 31: OFF waveform index

ex) nCP : number of control points  = 4

CP : control point coordinates : { 0,0,1,40,2,40,3,0}




3-4. spitting

bool selectSpittingNozzle(int nz, bool onoff) : nz = 0 ~ 15

bool startSpitting(double kHz, int nDrop = 0)

kHz = 0.1 ~ 50.0, nDrop = 0 : infinite, 1~32,767 : finite & auto stop

bool isSpittingStopped() : return true when stopped automatically (nDrop : finite)

bool stopSpitting() : You should call this whether the spitting was finite or infinite mode.

3-5. printing

a. bool beginWritePixel()

preparation for new printing data

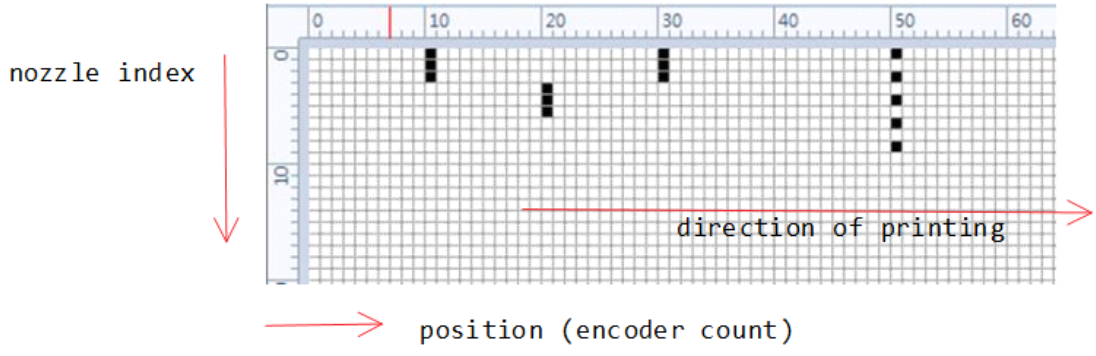
b. bool writePixel(int nPixel, int[] PixelPosition, byte[] PixelData)

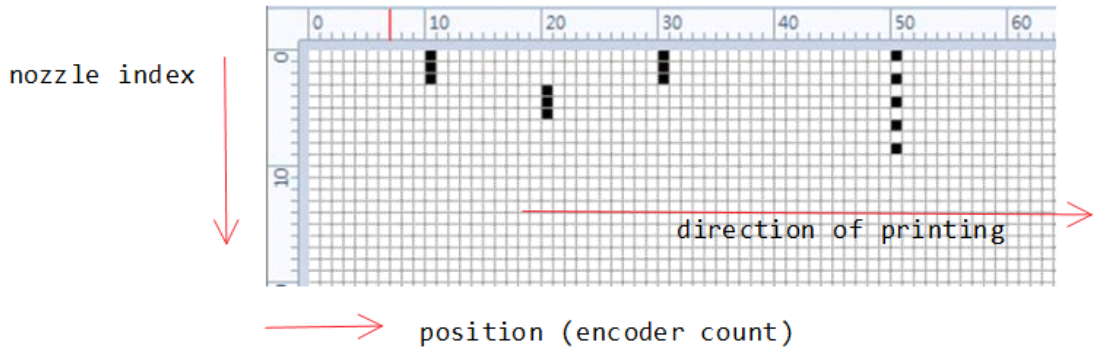
nPixel : number of the pixels to print

PixelPosition: array of pixel positions (encoder count value)

PixelData : onoff data for each nozzles (see below)

Example) print below image with a head of 10 nozzles






nPixel = 4 pixels

PixelPosition = { 10, 20, 30, 50}  // in the printing order

This print head has 10 nozzles.

PixelData = {  1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0 }

c. bool writePixelFinished()

wait for completion of writePixel

d. bool startPrinting()

call startPrinting() ,move the head carriage and image printed automatically.

e. bool stopPrinting()

After carriage movement finished, call stopPrinting().

If any error occurred during printing, it will return false.

Then you should check the error message RD.ErrorMessage.

3-6. heater

double getHeadTemperature() : return temperature of Dimatix DMC(Samba) cartridge

bool enableHeadHeater(double temperature)

bool disableHeadHeater()

3-7. trigger out

During spitting, trigger signal for LED stroboscope is generated and out at DSUB connector.

bool setTriggerOut(double delay,double duration)

delay after spitting trigger (us)

duration time (us)

bool setTriggerOutInverse(bool activeHigh)

# DPN16 C/C++ dll

1. DPN16 C-style DLL
   required files :

   RJETDLL.h // C-style header
   RJETDLL.dll // C-style DLL
   RDriverWrapperForC.dll // wrapper : class library -> C-style DLL
   RDriver.dll // class library
   XCOM.dll, WinUSBNet.dll // RDriver.dll depends on these files

2. Usage

   char* RJETgetErrorMessage(void) : error message
   return 1 when successful call.

   * headindex : not used, just place holder

3. Methods
   3-1. initalization

      int RJETinit(int id);

      low = 100 + rotary dip switch value (0 ~ 15)
      IP address = 192.168.0.low

      ex) rotary dip switch =0, Ethernet
      RJETinit(0) // ip= 192.168.0.100

   3-2. position encoder

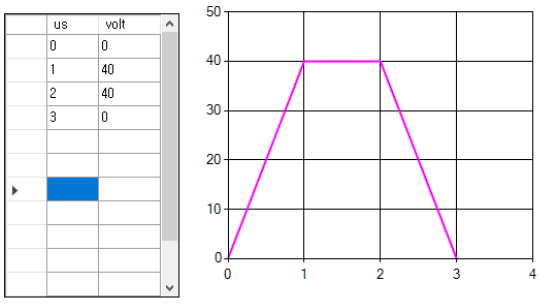
      int RJETresetPosition(int headindex) : reset encoder counter as 0
      int RJETgetPosition(int headindex) : get encoder counter

   3-3. waveform

int RJETsetFG(int headindex, int channel, int nCP, double CP[]);

channel : 0 ~ 15 : ON waveform index
16 ~ 31: OFF waveform index

ex) nCP : number of control points = 4
CP : control point coordinates : { 0,0,1,40,2,40,3,0}




3-4. spitting

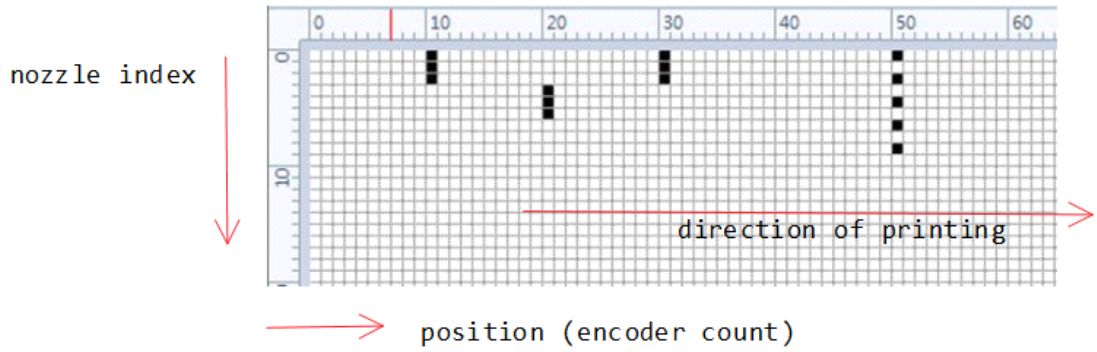
int RJETselectSpittingNozzle(int headindex, int nz, int onoff) : nz = 0 ~ 15
* int RJETenableSpitting(int headindex, int activeWave) : not used, just place holder
iint RJETstartSpitting(double kHz, int nDrop):
kHz = 0.1 ~ 50.0, nDrop = 0 : infinite, 1~32,767 : finite & auto stop

int RJETisSpittingStopped(void) : return true when stopped automatically (nDrop : finite)
int RJETstopSpitting(void) : You should call this whether the spitting was finite or infinite mode.

3-5. printing

a.int RJETbeginWritePixel(int headindex) : preparation for new printing data
b. int RJETwritePixel(int headindex, int nPixel, int PixelPosition[], unsigned char PixelData[])
nPixel : number of the pixels to print
PixelPosition: array of pixel positions (encoder count value)
PixelData : onoff data for each nozzles (see below)

Example) print below image with a head of 10 nozzles




nPixel = 4 pixels

PixelPosition = { 10, 20, 30, 50}  // in the printing order

This print head has 10 nozzles.

PixelData = {  1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0 }

c. int RJETendWritePixel(int headindex)

wait for completion of writePixel

d. int RJETstartPrinting(int headindex)

call startPrinting() ,move the head carriage and image printed automatically.

e. int RJETstopPrinting(int headindex)

After carriage movement finished, call stopPrinting().

If any error occurred during printing, it will return false.

Then you should check the error message RD.ErrorMessage.

3-6. heater

int RJETenableHeater(int headindex);

int RJETdisableHeater(int headindex);

int RJETsetHeater(int headindex, int temperature);

int RJETgetHeater(int headindex);

3-7. trigger out

During spitting, trigger signal for LED stroboscope is generated and out at DSUB connector.

int RJETsetTriggerOut(double delay,double duration)

delay after spitting trigger (us)

duration time (us)